
NLSY97 CODEBOOK SUPPLEMENT

MAIN FILE ROUND 1

Prepared for the
U.S. Department of Labor by

Center for Human Resource Research
The Ohio State University
1999

NLSY97 CODEBOOK SUPPLEMENT

MAIN FILE ROUND 1

**Prepared for the
U.S. Department of Labor by**

**Center for Human Resource Research
The Ohio State University
1999**

This publication is prepared in conjunction with contract #J-9-J-5-0030 with the Bureau of Labor Statistics, U.S. Department of Labor. None of its contents is to be construed as necessarily representing the official position or policy of the Department of Labor.

Table of Contents

Introduction to the CAPI Questionnaire and Codebook.....	i
NLSY97 Attachment 1: 1990 Census Industrial & Occupational Classification Codes	1
Introduction to the Created Variable Appendices.....	23
NLSY97 Appendix 1: Education Variable Creation	27
NLSY97 Appendix 2: Employment Variable Creation	47
NLSY97 Appendix 3: Family Background Variable Creation	101
NLSY97 Appendix 4: Geographic Variable Creation	139
NLSY97 Appendix 5: Income and Assets Variable Creation	143
NLSY97 Appendix 6: Event History Creation and Documentation	189
NLSY97 Appendix 7: Continuous Month Scheme and Crosswalk	197
NLSY97 Appendix 8: Round 1 Instrument Rosters.....	207

Also considered part of the Codebook Supplement is Appendix 9, "Family Process and Adolescent Outcome Measures." This appendix, which describes a number of variables created by Child Trends, Inc., is provided in a separate printed document.

For more information about any aspect of the NLS program, contact:

NLS User Services
Center for Human Resource Research
921 Chatham Lane, Suite 100
Columbus, OH 43221-2418
(614) 442-7366
usersvc@postoffice.chrr.ohio-state.edu

Introduction to the CAPI Questionnaire and Codebook

The newest survey in the NLS program, the National Longitudinal Survey of Youth 1997 (NLSY97) is designed to be representative of the U.S. population born during the years 1980 through 1984. Through the NLSY97, the Bureau of Labor Statistics (BLS) will be able to identify characteristics that define the transition today's youths make from school to the labor market and into adulthood. The NLSY97 cohort includes 8,984 respondents ages 12–16 as of December 31, 1996; the sample contains a cross-sectional subsample and an oversample of black and Hispanic respondents. More information on the selection of respondents for the cohort is available in the *NLSY97 User's Guide*.

This survey was conducted as a computer-assisted personal interview (CAPI). The Round 1 survey was conducted using three different questionnaires: the *Screener, Household Roster, and Non-Resident Roster Questionnaire*; the *Youth Questionnaire*; and the *Parent Questionnaire*. The *Screener, Household Roster, and Non-Resident Roster Questionnaire* was administered to a household resident over the age of 18. The *Youth Questionnaire* was administered to the youth respondent living in the household. The *Parent Questionnaire* was administered to a parent or parent-like figure of the youth residing in the household. For information about how the responding parent was selected, researchers should refer to the *NLSY97 User's Guide*. The content of these survey instruments is presented in separate questionnaire documents, available from NLS User Services.

Frequencies from the data collected during the NLSY97 interviews are provided for researchers in the form of a codebook contained on the NLSY97 CD-ROM. For each variable taken directly from the interview, the codebook provides information about the question or check item, the variable reference number, the universe of respondents to whom the question applied, and the distribution of responses to the question. The codebook also contains a number of variables created by survey personnel after the interview, providing information about the content of the variable and the distribution of responses.

Some information relevant to the data contained in the codebook cannot be included on the CD-ROM due to practical constraints. This document, the *NLSY97 Codebook Supplement*, presents additional information which will help researchers to use the data more effectively. The first attachment lists industry and occupation codes and their associated descriptors; this list is too lengthy to be incorporated into the codebook. A number of appendices then provide programs for the created variables; researchers may want to use this information to identify the raw survey data used in a created variable and to understand the rules applied during the creation process. This document also describes the creation of the event history arrays and helps researchers to understand their structure and use.

This introduction to the *Codebook Supplement* is intended to assist researchers in understanding some of the terms and survey methods associated with a CAPI interview. The way in which a CAPI instrument is constructed has important implications for the presentation of the data, and so this discussion will aid in interpretation of the codebook and accompanying documentation.

I. TERMS

Discussions of CAPI surveys include a number of terms that may be unfamiliar to many researchers. The following terms are used throughout the survey documentation; we provide definitions here for easy reference.

Instrument Rosters

A “roster” is a list of one or more items of information pertaining to a specific set of subjects, such as the biological children of the respondent or members of the respondent’s household. For instance, the BIOCHILD roster contains a variety of items pertaining to each biological child of the respondent, (e.g., name, gender, birthdate, etc.). By using the roster format, the CAPI program can gather an inventory of information, tag the data to a specific subject, carry the data along through the interview, and access them when necessary. This format also allows for these items of information to be presented to the interviewer

at any time. A listing of rosters, such as the Household Roster, is included in this *Codebook Supplement* as appendix 8. This listing includes the contents of the rosters, the applicable names attached to the rosters that might be encountered in the codebook and/or the questionnaire, and the variable reference number assigned to each piece of data in the codebook.

Interviewer's Reference Manual

The *Interviewer's Reference Manual* reproduces the electronic "help screens" that were available to the interviewers for specific questions. These help screens contain definitions, instructions, and other information which interviewers used during the interview to obtain consistent information from respondents, as well as listings of the questions for which they were used during the interview. The *Interviewer's Reference Manual* should be used in conjunction with the codebook or the questionnaire so that researchers can fully understand the intent of each question in the survey.

Symbols and "Text Fills"

Symbols are reserved fields into which data can be placed, stored and accessed throughout the questionnaire. Each symbol field is assigned a distinct name, often with an index number appended when the same piece of information is stored for a set of subjects. For instance, the symbol "lintdate" contains the date of last interview for the respondent. The symbols "emp.name(1)" through "emp.name(7)" contain the names of the first through the seventh employers of the respondent. Throughout the survey, the last interview date and/or the name of the respondent's first employer can be accessed by invoking the symbol names "lintdate" and "emp.name(1)" respectively.

Users will encounter these symbol names both in the codebook and in the questionnaire. Sometimes data in a symbol is accessed and used to govern a skip or perform a calculation. At other times, a symbol name may be part of a question text. In this case, the content of the specific symbol content becomes part of the text of the question, and is read as such. This is referred to as a "text substitution" or "text fill." For instance, a question text reading "When you started working for [emp.name(1)], what kind of work did you do? That is, what was your occupation?" would appear to the interviewer with the appropriate name replacing the symbol "emp.name(1)."

Other types of text fills are also present in the questionnaire. For example, the responding parent may be asked the following question: "What was the reason [he/she] did not live with [his/her] [mother/father/parents]?" In this question, "he/she" and "his/her" refer to the parent's spouse or partner. The computer automatically fills in the correct gender, and the interviewer reads the question appropriately. "Mother/father/parents" in the question text indicates a set of possible responses to the previous question. The computer automatically fills in the appropriate choice from the three words for the given interview. These automated text fills reduce error and remove the burden of asking the question using the appropriate phrasing from the interviewer.

Loops

Certain sequences of questions in the CAPI instrument are repeated a number of times. For instance, some sets of questions are repeated up to 7 times in the Employment Section of the youth questionnaire, for each of up to 7 jobs. Question names that include a ".01", ".02", ".03", etc. at the end, belong to these repeating sequences of questions. Each repetition of the sequence of questions is referred to as a "loop." Taking the Employment Section as an example, the sequence of questions asking about the first job is referred to as the first "loop." The sequence asking about the second employer is referred to as the second "loop," and so on. While the codebook generally includes more than one loop in a series (because more than one may contain valid data), the questionnaire includes only the **first loop** in each set of loops.

Hard and Soft Range Restrictions

The "Hard Minimum," "Hard Maximum," "Soft Minimum," and "Soft Maximum" specifications control the allowable range of values that can be entered for a given question. These fields are only active when a

question calls for the entry of a time date or amount. Questions that require the interviewer to select one response or to select all that apply do not contain this field, as the range limits are implicit in the distribution code block and are thereby enforced.

Hard minima and maxima are absolute limits that an interviewer- or respondent-generated answer must obey. Entry of values outside the hard range is not allowed. In such cases the interviewer is instructed to enter the maximum or minimum allowable value, as appropriate, enter the actual response in the comment field, and “flag” the case for central office checking. Soft minima and maxima are nested within the hard range. When a response falls outside the soft range but inside the hard range, the computer beeps and asks the interviewer to either confirm or change the response.

In some cases, the hard ranges are themselves determined by a variable. For example, a question may use the respondent’s birth date as a minimum and the current interview date as a maximum, preventing the interviewer from entering a date earlier than the hard minimum birth date or later than the hard maximum interview date. This is a powerful tool for the collection of event histories (among other types of data sequences) and is used extensively in the instrument.

II. APPEARANCE AND PRESENTATION OF DATA

Some CAPI questions generate more than one variable. For example, some questions collect information about the date an event happened and generate three variables: month, day and year. Similarly, questions that ask the respondent to indicate which of several responses are appropriate, and to pick all responses that apply, can generate multiple variables. When a question record generates multiple variables, those variables have decimals in the reference numbers. These two types of questions (date-entry and “code-all-that-apply”) and the examples of the resulting variables, are discussed further below.

Date-Entry Questions

All date questions, whether full dates (month, day and year) or just month and year, are represented in the CAPI codebook with, respectively, three or two variables. Each variable contains the same codeblock displaying the ranges and missing values for all elements of the date. A base reference number ending in “.00”, is assigned to the first variable in the set. The same base reference number, with endings of “.01” and “.02” if necessary, is assigned to the other elements of the date. Thus, if the day is assigned a reference number of R10851.00 for the variable “Date of Birth of HH Member 01 (Scr Ros Item),” the month and year would be assigned reference numbers of R10851.01 and R10851.02 respectively.

Code-All-That-Apply Questions

The NLSY97 includes questions that allow respondents to give multiple responses or code all responses that apply. In the CAPI codebook, each possible response constitutes its own variable. In each codeblock for a given code-all-that-apply question, frequencies for all possible responses are represented. However, each specific variable contains **only** the valid data for one specific possible response, as researchers will note when extracting data. Reference numbers are assigned in the same manner as described for date-entry questions. A base reference number is assigned to the first possible response, with a decimal value being appended to that base number for each following possible response. For example, variables R02461.00–R02461.11 (Activities to Find a Job Emp 01) provide responses (coded yes/no) for 12 different methods of finding a job. Although the codeblock for R02461.01 represents the frequencies for all possible responses to the question, accessing the data for that specific variable will produce only the data for the response “Contacted employment agency.”

Machine-Generated Check Items

Many questions in the NLSY97 data are “machine checks.” In these questions, previously reported information is checked by the computer and computations are made automatically, causing the

appropriate skip pattern to be executed without intervention by the interviewer. For example, the survey program may check the respondent's gender before asking women whether they have ever been pregnant and men whether they have ever fathered a child. In an effort to clarify the skip patterns present in the instrument, a large number of these machine checks appear in the codebook. The text of machine checks is generally in machine language or equation form; the codeblock also includes a note clarifying the purpose of the check. For example, R02678., "Chk R Female (Pregnancy Leave) Emp 01," includes the following code:

([male/female] = 2);

The codeblock also contains a translation of the code which is more accessible to users:

/* Is respondent a female? This determines whether the paid pregnancy leave questions which follow are asked. */

NLSY97 Attachment 1:
1990 Census Industrial & Occupational
Classification Codes

U.S. DEPARTMENT OF COMMERCE
Bureau of the Census
Washington, DC 20233

1990 CENSUS OF POPULATION INDUSTRIAL CLASSIFICATION SYSTEM

“N.e.c.” means not elsewhere classified.

1990 Census Code	Industry
AGRICULTURE, FORESTRY, AND FISHERIES	
010	Agricultural production, crops
011	Agricultural production livestock
012	Veterinary services
020	Landscape and horticultural services
030	Agricultural services, n.e.c.
031	Forestry
032	Fishing, hunting, and trapping
MINING	
040	Metal mining
041	Coal mining
042	Oil and gas extraction
050	Nonmetallic mining and quarrying, except fuel
060	CONSTRUCTION
MANUFACTURING	
Nondurable Goods	
<i>Food and kindred products</i>	
100	Meat products
101	Dairy products
102	Canned, frozen, and preserved fruits and vegetables
110	Grain mill products
111	Bakery products
112	Sugar and confectionery products
120	Beverage industries
121	Miscellaneous food preparations and kindred products
122	Not specified food industries
130	Tobacco manufactures
<i>Textile mill products</i>	
132	Knitting mills
140	Dyeing and finishing textiles, except wool and knit goods
141	Carpets and rugs
142	Yarn, thread, and fabric mills
150	Miscellaneous textile mill products
<i>Apparel and other finished textile products</i>	
151	Apparel and accessories, except knit
152	Miscellaneous fabricated textile products
<i>Paper and allied products</i>	
160	Pulp, paper, and paperboard mills
161	Miscellaneous paper and pulp products
162	Paperboard containers and boxes

1990 Census Code	Industry
	<i>Printing, publishing, and allied industries</i>
171	Newspaper publishing and printing
172	Printing, publishing and allied industries, except newspapers
	<i>Chemicals and allied products</i>
180	Plastics, synthetics, and resins
181	Drugs
182	Soaps and cosmetics
190	Paints, varnishes, and related products
191	Agricultural chemicals
192	Industrial and miscellaneous chemicals
	<i>Petroleum and coal products</i>
200	Petroleum refining
201	Miscellaneous petroleum and coal products
	<i>Rubber and miscellaneous plastics products</i>
210	Tires and inner tubes
211	Other rubber products, and plastics footwear and belting
212	Miscellaneous plastics products
	<i>Leather and leather products</i>
220	Leather tanning and finishing
221	Footwear, except rubber and plastic
222	Leather products, except footwear
	Durable Goods
	<i>Lumber and wood products, except furniture</i>
230	Logging
231	Sawmills, planing mills, and millwork
232	Wood buildings and mobile homes
241	Miscellaneous wood products
242	Furniture and fixtures
	<i>Stone, clay, glass, and concrete products</i>
250	Glass and glass products
251	Cement, concrete, gypsum, and planter products
252	Structural clay products
261	Pottery and related products
262	Miscellaneous nonmetallic mineral and stone products
	<i>Metal industries</i>
270	Blast furnaces, steelworks, rolling and finishing mills
271	Iron and steel foundries
272	Primary aluminum industries
280	Other primary metal industries
281	Cutlery, handtools, and general hardware
282	Fabricated structural metal products
292	Ordnance
300	Miscellaneous fabricated metal products
301	Not specified metal industries

1990 Census Code	Industry
	<i>Machinery and computing equipment</i>
310	Engines and turbines
311	Farm machinery and equipment
312	Construction and material handling machines
320	Metalworking machinery
321	Office and accounting machines
322	Computers and related equipment
331	Machine, except electrical, n.e.c.
332	Not specified machinery
	<i>Electrical machinery, equipment, and supplies</i>
340	Household appliances
341	Radio, TV, and communication equipment
342	Electrical machinery, equipment, and supplies, n.e.c.
350	Not specified electrical machinery, equipment, and supplies
	<i>Transportation equipment</i>
351	Motor vehicles and motor vehicle equipment
352	Aircraft and parts
360	Ship and boat building and repairing
361	Railroad locomotives and equipment
362	Guided missiles, space vehicles, and parts
370	Cycles and miscellaneous transportation equipment
	<i>Professional and photographic equipment, and watches</i>
371	Scientific and controlling instruments
372	Medical, dental, and optical instruments and supplies
380	Photographic equipment and supplies
381	Watches, clocks, and clockwork operated devices
390	Toys, amusement, and sporting goods
391	Miscellaneous manufacturing industries
392	Not specified manufacturing industries
TRANSPORTATION, COMMUNICATIONS AND OTHER PUBLIC UTILITIES	
	Transportation
400	Railroads
401	Bus service and urban transit
402	Taxicab service
410	Trucking service
411	Warehousing and storage
412	U.S. Postal Service
420	Water transportation
421	Air transportation
422	Pipelines, except natural gas
432	Services incidental to transportation
	Communications
440	Radio and television broadcasting and cable
441	Telephone communications
442	Telegraph and miscellaneous communications services

1990 Census Code	Industry
	Utilities and sanitary services
450	Electric light and power
451	Gas and steam supply systems
452	Electric and gas, and other combinations
470	Water supply and irrigation
471	Sanitary services
472	Not specified utilities
	WHOLESALE TRADE
	Durable Goods
500	Motor vehicles and equipment
501	Furniture and home furnishings
502	Lumber and construction materials
510	Professional and commercial equipment and supplies
511	Metals and minerals, except petroleum
512	Electrical goods
521	Hardware, plumbing and heating supplies
530	Machinery, equipment, and supplies
531	Scrap and waste materials
532	Miscellaneous wholesale, durable goods
	Nondurable Goods
540	Paper and paper products
541	Drugs, chemicals and allied products
542	Apparel, fabrics, and notions
550	Groceries and related products
551	Farm-product raw materials
552	Petroleum products
560	Alcoholic beverages
561	Farm supplies
562	Miscellaneous wholesale, nondurable goods
571	Not specified wholesale trade
	RETAIL TRADE
580	Lumber and building material retailing
581	Hardware stores
582	Retail nurseries and garden stores
590	Mobile home dealers
591	Department stores
592	Variety stores
600	Miscellaneous general merchandise stores
601	Grocery stores
602	Dairy products stores
610	Retail bakeries
611	Food stores, n.e.c.
612	Motor vehicle dealers
620	Auto and home supply stores
621	Gasoline service stations
622	Miscellaneous vehicle dealers
623	Apparel and accessory stores, except shoe
630	Shoe stores
631	Furniture and home furnishings stores

1990 Census Code	Industry
632	Household appliance stores
633	Radio, TV, and computer stores
640	Music stores
641	Eating and drinking places
642	Drug stores
650	Liquor stores
651	Sporting goods, bicycles, and hobby stores
652	Book and stationary stores
660	Jewelry stores
661	Gift, novelty, and souvenir shops
662	Sewing, needlework and piece goods stores
663	Catalog and mail order houses
670	Vending machine operators
671	Direct selling establishments
672	Fuel dealers
681	Retail florists
682	Miscellaneous retail stores
691	Not specified retail trade
FINANCE, INSURANCE, AND REAL ESTATE	
700	Banking
701	Savings institutions, including credit unions
702	Credit agencies, n.e.c.
710	Security, commodity brokerage, and investment companies
711	Insurance
712	Real estate, including real estate-insurance offices
BUSINESS AND REPAIR SERVICES	
721	Advertising
722	Services to dwellings and other buildings
731	Personnel supply services
732	Computer and data processing services
740	Detective and protective services
741	Business services, n.e.c.
742	Automotive rental and leasing, without drivers
750	Automobile parking and carwashes
751	Automotive repair and related services
752	Electrical repair shops
760	Miscellaneous repair services
PERSONAL SERVICES	
761	Private households
762	Hotels and motels
770	Lodging places, except hotels and motels
771	Laundry, cleaning, and garment services
772	Beauty shops
780	Barber shops
781	Funeral service and crematories
782	Shoe repair shops
790	Dressmaking shops
791	Miscellaneous personal services

1990 Census Code

Industry

ENTERTAINMENT AND RECREATION SERVICES

- | | |
|-----|---|
| 800 | Theaters and motion pictures |
| 801 | Video tape rental |
| 802 | Bowling centers |
| 810 | Miscellaneous entertainment and recreation services |

PROFESSIONAL AND RELATED SERVICES

- | | |
|-----|---|
| 812 | Offices and clinics of physicians |
| 820 | Offices and clinics of dentists |
| 821 | Offices and clinics of chiropractors |
| 822 | Offices and clinics of optometrists |
| 830 | Offices and clinics of health practitioners, n.e.c. |
| 831 | Hospitals |
| 832 | Nursing and personal care facilities |
| 840 | Health services, n.e.c. |
| 841 | Legal services |
| 842 | Elementary and secondary schools |
| 850 | Colleges and universities |
| 851 | Vocational schools |
| 852 | Libraries |
| 860 | Educational services, n.e.c. |
| 861 | Job training and vocational rehabilitation services |
| 862 | Child day care services |
| 863 | Family child care homes |
| 870 | Residential care facilities, without nursing |
| 871 | Social services, n.e.c. |
| 872 | Museums, art galleries and zoos |
| 873 | Labor unions |
| 880 | Religious organizations |
| 881 | Membership organizations, n.e.c. |
| 882 | Engineering architectural and surveying services |
| 890 | Accounting, auditing, and bookkeeping services |
| 891 | Research, development, and testing services |
| 892 | Management and public relations services |
| 893 | Miscellaneous professional and related services |

PUBLIC ADMINISTRATION

- | | |
|-----|--|
| 900 | Executive and legislative offices |
| 901 | General government, n.e.c. |
| 910 | Justice, public order, and safety |
| 921 | Public finance, taxation, and monetary policy |
| 922 | Administration of human resources programs |
| 930 | Administration of environmental quality and housing programs |
| 931 | Administration of economic programs |
| 932 | National security and international affairs |
| 991 | Military |

U.S. DEPARTMENT OF COMMERCE
Bureau of the Census
Washington, DC 20233

1990 CENSUS OF POPULATION OCCUPATIONAL CLASSIFICATION SYSTEM

"N.e.c." means not elsewhere classified.

1990 Census Code	Industry
MANAGERIAL AND PROFESSIONAL SPECIALTY OCCUPATIONS	
Executive, Administrative, and Managerial Occupations	
003	Legislators
004	Chief executives and general administrators, public administration
005	Administrators and officials, public administration
006	Administrators, protective services
007	Financial managers
008	Personnel and labor relations managers
009	Purchasing managers
013	Managers, marketing, advertising, and public relations
014	Administrators, education and related fields
015	Managers, medicine and health
016	Postmasters and mail superintendents
017	Managers, food serving and lodging establishments
018	Managers, properties and real estate
019	Funeral directors
021	Managers, service organizations, n.e.c.
022	Managers and administrators n.e.c.
Management Related Occupations	
023	Accountants and auditors
024	Underwriters
025	Other financial officers
026	Management analysts
027	Personnel, training, and labor relations specialists
028	Purchasing agents and buyers, farm products
029	Buyers, wholesale and retail trade except farm products
033	Purchasing agents and buyers, n.e.c.
034	Business and promotion agents
035	Construction inspectors
036	Inspectors and compliance officers, except construction
037	Management related occupations, n.e.c.
PROFESSIONAL SPECIALTY OCCUPATIONS	
Engineers, Architects, and Surveyors	
043	Architects
	<i>Engineers</i>
044	Aerospace
045	Metallurgical and materials
046	Mining
047	Petroleum
048	Chemical
049	Nuclear
053	Civil
054	Agricultural

1990 Census Code	Industry
055	Electrical and electronic
056	Industrial
057	Mechanical
058	Marine and naval architects
059	Engineers, n.e.c.
063	Surveyors and mapping scientists
	<i>Mathematical and Computer Scientists</i>
064	Computer systems analysts and scientists
065	Operations and systems researchers and analysts
066	Actuaries
067	Statisticians
068	Mathematical scientists, n.e.c.
	<i>Natural Scientists</i>
069	Physicists and astronomers
073	Chemists, except biochemists
074	Atmospheric and space scientists
075	Geologists and geodesists
076	Physical scientists, n.e.c.
077	Agricultural and food scientists
078	Biological and life scientists
079	Forestry and conservation scientists
083	Medical scientists
	<i>Health Diagnosing Occupations</i>
084	Physicians
085	Dentists
086	Veterinarians
087	Optometrists
088	Podiatrists
089	Health diagnosing practitioners, n.e.c.
	<i>Health Assessment and Treating Occupations</i>
095	Registered nurses
096	Pharmacists
097	Dietitians
	<i>Therapists</i>
098	Respiratory therapists
099	Occupational therapists
103	Physical therapists
104	Speech therapists
105	Therapists, n.e.c.
106	Physicians' assistants
	<i>Teachers, Postsecondary</i>
113	Earth, environmental, and marine science teachers
114	Biological science teachers
115	Chemistry teachers
116	Physics teachers
117	Natural science teachers, n.e.c.

1990 Census Code	Industry
118	Psychology teachers
119	Economics teachers
123	History teachers
124	Political science teachers
125	Sociology teachers
126	Social science teachers, n.e.c.
127	Engineering teachers
128	Mathematical science teachers
129	Computer science teachers
133	Medical science teachers
134	Health specialties teachers
135	Business, commerce, and marketing teachers
136	Agriculture and forestry teachers
137	Art, drama, and music teachers
138	Physical education teachers
139	Education teachers
143	English teachers
144	Foreign language teachers
145	Law teachers
146	Social work teachers
147	Theology teachers
148	Trade and industrial teachers
149	Home economics teachers
153	Teachers, postsecondary, n.e.c.
154	Postsecondary teachers, subject not specified
	<i>Teachers, Except Postsecondary</i>
155	Teachers, prekindergarten and kindergarten
156	Teachers, elementary school
157	Teachers, secondary school
158	Teachers, special education
159	Teachers, n.e.c.
163	Counselors, educational and vocational
	<i>Librarians, Archivists, and Curators</i>
164	Librarians
165	Archivists and curators
	<i>Social Scientists and Urban Planners</i>
166	Economists
167	Psychologists
168	Sociologists
169	Social scientists, n.e.c.
173	Urban planners
	<i>Social, Recreation, and Religious Workers</i>
174	Social workers
175	Recreation workers
176	Clergy
177	Religious workers, n.e.c.

1990 Census Code	Industry
	<i>Lawyers and Judges</i>
178	Lawyers
179	Judges
	<i>Writers, Artists, Entertainers, and Athletes</i>
183	Authors
184	Technical writers
185	Designers
186	Musicians and composers
187	Actors and directors
188	Painters, sculptors, craft-artists, and artist printmakers
189	Photographers
193	Dancers
194	Artists, performers, and related workers, n.e.c.
195	Editors and reporters
197	Public relations specialists
198	Announcers
199	Athletes

TECHNICAL, SALES AND ADMINISTRATIVE SUPPORT OCCUPATIONS

Technicians and Related Support occupations

Health Technologists and Technicians

203	Clinical laboratory technologists and technicians
204	Dental hygienists
205	Health record technologists and technicians
206	Radiologic technicians
207	Licensed practical nurses
208	Health technologists and technicians, n.e.c.

Technologists and Technicians, Except Health

Engineering and Related Technologists and Technicians

213	Electrical and electronic technicians
214	Industrial engineering technicians
215	Mechanical engineering technicians
216	Engineering technicians, n.e.c.
217	Drafting occupations
218	Surveying and mapping technicians

Science Technicians

223	Biological technicians
224	Chemical technicians
225	Science technicians

Technicians: Except Health, Engineering and Science

226	Airplane pilots and navigators
227	Air traffic controllers
228	Broadcast equipment operators
229	Computer programmers
233	Tool programmers, numerical control
234	Legal assistants
235	Technicians, n.e.c.

1990 Census Code	Industry
	Sales Occupations
243	Supervisors and proprietors, sales occupations
	<i>Sales Representatives, Finance and Business Services</i>
253	Insurance sales occupations
254	Real estate sales occupations
255	Securities and financial services sales occupations
256	Advertising and related sales occupations
257	Sales occupations, other business services
	<i>Sales Representatives, Commodities Except Retail</i>
258	Sales engineers
259	Sales representatives, mining, manufacturing, and wholesale
	<i>Sales Workers, Retail and Personal Services</i>
263	Sales workers, motor vehicles and boats
264	Sales workers, apparel
265	Sales workers, shoes
266	Sales workers, furniture and home furnishings
267	Sales workers; radio, TV, hi-fi, and appliances
268	Sales workers, hardware and building supplies
269	Sales workers, parts
274	Sales workers, other commodities
275	Sales counter clerks
276	Cashiers
277	Street and door-to-door sales workers
278	News vendors
	<i>Sales Related Occupations</i>
283	Demonstrators, promoters and models, sales
284	Auctioneers
285	Sales support occupations, n.e.c.
	Administrative Support Occupations, Including Clerical
	<i>Supervisors, Administrative Support Occupations</i>
303	Supervisors, general office
304	Supervisors, computer equipment operators
305	Supervisors, financial records processing
306	Chief communications operators
307	Supervisors; distribution, scheduling, and adjusting clerks
	<i>Computer Equipment Operators</i>
308	Computer operators
309	Peripheral equipment operators
	<i>Secretaries, Stenographers, and Typists</i>
313	Secretaries
314	Stenographers
315	Typists
	<i>Information Clerks</i>
316	Interviewers
317	Hotel clerks

1990 Census Code	Industry
318	Transportation ticket and reservation agents
319	Receptionists
323	Information clerks, n.e.c.
	<i>Records Processing Occupations, Except Financial</i>
325	Classified-ad clerks
326	Correspondence clerks
327	Order clerks
328	Personnel clerks, except payroll and timekeeping
329	Library clerks
335	File clerks
336	Records clerks
	<i>Financial Records Processing Occupations</i>
337	Bookkeepers, accounting, and auditing clerks
338	Payroll and timekeeping clerks
339	Billing clerks
343	Cost and rate clerks
344	Billing, posting, and calculating machine operators
	<i>Duplicating, Mail and Other Office Machine Operators</i>
345	Duplicating machine operators
346	Mail preparing and paper handling machine operators
347	Office machine operators, n.e.c.
	<i>Communications Equipment Operators</i>
348	Telephone operators
353	Communications equipment operators, n.e.c.
	<i>Mail and Message Distributing Occupations</i>
354	Postal clerks, exc. mail carriers
355	Mail carriers, postal service
356	Mail clerks, exc. postal service
357	Messengers
	<i>Material Recording, Scheduling, and Distributing Clerks</i>
359	Dispatchers
363	Production coordinators
364	Traffic, shipping, and receiving clerks
365	Stock and inventory clerks
366	Motor readers
368	Weighers, measurers, checkers and samplers
373	Expeditors
374	Material recording, scheduling, and distributing clerks, n.e.c.
	<i>Adjusters and Investigators</i>
375	Insurance adjusters, examiners, and investigators
376	Investigators and adjusters except insurance
377	Eligibility clerks, social welfare
378	Bill and account collectors
	<i>Miscellaneous Administrative Support Occupations</i>
379	General office clerks

1990 Census Code	Industry
383	Bank tellers
384	Proofreaders
385	Data-entry keyers
386	Statistical clerks
387	Teachers' aides
389	Administrative support occupations, n.e.c.

SERVICE OCCUPATIONS

Private Household Occupations

403	Launderers and ironers
404	Cooks, private household
405	Housekeepers and butlers
406	Child care workers, private household
407	Private household cleaners and servants

Protective Service Occupations

Supervisors Protective Service Occupations

413	Supervisors, firefighting and fire prevention occupations
414	Supervisors, police and detectives
415	Supervisors, guards

Firefighting and Fire Prevention Occupations

416	Fire inspection and fire prevention occupations
417	Firefighting occupations

Police and Detectives

418	Police and detectives, public service
423	Sheriffs, bailiffs, and other law enforcement officers
424	Correctional institution officers

Guards

425	Crossing guards
426	Guards and police, except public service
427	Protective service occupations, n.e.c.

Service Occupations, Except Protective and Household

Food Preparation and Service Occupations

433	Supervisors, food preparation and service occupations
434	Bartenders
435	Waiters and waitresses
436	Cooks
438	Food counter, fountain and related occupations
439	Kitchen workers, food preparation
443	Waiters'/waitresses' assistants
444	Miscellaneous food preparation occupations

Health Service Occupations

445	Dental assistants
446	Health aides, except nursing
447	Nursing aides, orderlies, and attendants

Cleaning and Building Service Occupations, except Household

448	Supervisors cleaning and building service workers
-----	---

1990 Census Code	Industry
449	Maids and houseman
453	Janitors and cleaners
454	Elevator operators
455	Post control occupations
	<i>Personal Service Occupations</i>
456	Supervisors, personal service occupations
457	Barbers
458	Hairdressers and cosmetologists
459	Attendants, amusement and recreation facilities
461	Guides
462	Ushers
463	Public transportation attendants
464	Baggage porters and bellhops
465	Welfare service aides
466	Family child care providers
467	Early childhood teacher's assistants
468	Child care workers, n.e.c.
469	Personal service occupations, n.e.c.
	FARMING, FORESTRY AND FISHING OCCUPATIONS
	Farm Operators and Managers
473	Farmers, except horticultural
474	Horticultural specialty farmers
475	Managers, farms, except horticultural
476	Managers, horticultural specialty farms
	Other Agricultural and Related Occupations
	<i>Farm Occupations Except Managerial</i>
477	Supervisors, farm workers
479	Farm workers
483	Marine life cultivation workers
484	Nursery workers
	<i>Related Agricultural Occupations</i>
485	Supervisors related agricultural occupations
486	Groundskeepers and gardeners, except farm
487	Animal caretakers, except farm
488	Graders and sorters, agricultural products
489	Inspectors, agricultural products
	Forestry and Logging Occupations
494	Supervisors, forestry, and logging workers
495	Forestry workers, except logging
496	Timber cutting and logging occupations
	Fishers, Hunters, and Trappers
497	Captains and other officers, fishing vessels
498	Fishers
499	Hunters and trappers

1990 Census Code	Industry
PRECISION PRODUCTION, CRAFT, AND REPAIR OCCUPATIONS	
	Mechanics and Repairers
503	Supervisors, mechanics and repairers
	<i>Mechanics and Repairers, Except Supervisors</i>
	Vehicle and Mobile Equipment Mechanics and Repairers
505	Automobile mechanics
506	Automobile mechanic apprentices
507	Bus, truck, and stationary engine mechanics
508	Aircraft engine mechanics
509	Small engine repairers
514	Automobile body and related repairers
515	Aircraft mechanics, exc. engine
516	Heavy equipment mechanics
517	Farm equipment mechanics
518	Industrial machinery repairers
519	Machinery maintenance occupations
	Electrical and Electronic Equipment Repairers
523	Electronic repairers, communications and industrial equipment
525	Data processing equipment repairers
526	Household appliance and power tool repairers
527	Telephone line installers and repairers
529	Telephone installers and repairers
533	Miscellaneous electrical and electronic equipment repairers
534	Heating, air conditioning, and refrigeration mechanics
	Miscellaneous Mechanics and Repairers
535	Camera, watch, and musical instrument repairers
536	Locksmiths and safe repairers
538	Office machine repairers
539	Mechanical controls and valve repairers
543	Elevator installers and repairers
544	Millwrights
547	Specified mechanics and repairers, n.e.c.
549	Not specified mechanics and repairers
	Construction Trades
	<i>Supervisors, Construction Occupations</i>
553	Supervisors; brickmasons, stonemasons, and tile setters
554	Supervisors; carpenters and related workers
555	Supervisors; electricians and power transmission installers
556	Supervisors; painters, paperhangers, and plasterers
557	Supervisors; plumbers, pipefitters, and steamfitters
558	Supervisors; n.e.c.
	<i>Construction Trades Except Supervisors</i>
563	Brickmasons and stonemasons
564	Brickmason and stonemason apprentices
565	Tile setters, hard and soft
566	Carpet installers
567	Carpenters

1990 Census Code	Industry
569	Carpenter apprentices
573	Drywall installers
575	Electricians
576	Electrician apprentices
577	Electrical power installers and repairers
579	Painters, construction and maintenance
583	Paperhangers
584	Plasterers
585	Plumbers, pipefitters, and steamfitters
587	Plumber, pipefitter, and steamfitter apprentices
588	Concrete and terrazzo finishers
589	Glaziers
593	Insulation workers
594	Paving, surfacing, and tamping equipment operators
595	Roofers
596	Sheetmetal duct installers
597	Structural metal workers
598	Drillers, earth
599	Construction trades, n.e.c.
	Extractive Occupations
613	Supervisors, extractive occupations
614	Drillers, oil well
615	Explosives workers
616	Mining machine operators
617	Mining occupations, n.e.c.
	Precision Production Occupations
628	Supervisors, production occupations
	<i>Precision Metal Working Occupations</i>
634	Tool and die makers
635	Tool and die maker apprentices
636	Precision assemblers, metal
637	Machinists
639	Machinist apprentices
643	Boilermakers
644	Precision grinders, filers, and tool sharpeners
645	Patternmakers and model makers, metal
646	Lay-out workers
647	Precious stones and metals workers (Jewelers)
649	Engravers, metal
653	Sheet metal workers
654	Sheet metal worker apprentices
655	Miscellaneous precision metal workers
	<i>Precision Woodworking Occupations</i>
656	Patternmakers and model makers, wood
657	Cabinet makers and bench carpenters
658	Furniture and wood finishers
659	Miscellaneous precision woodworkers

1990 Census Code	Industry
	<i>Precision Textile, Apparel, and Furnishings Machine Workers</i>
666	Dressmakers
667	Tailors
668	Upholsterers
669	Shoe repairers
674	Miscellaneous precision apparel and fabric workers
	<i>Precision Workers, Assorted Materials</i>
675	Hand molders and shapers, except jewelers
676	Patternmakers, lay-out workers, and cutters
677	Optical goods workers
678	Dental laboratory and medical appliance technicians
679	Bookbinders
683	Electrical and electronic equipment assemblers
684	Miscellaneous precision workers, n.e.c.
	<i>Precision Food Production Occupations</i>
686	Butchers and meat cutters
687	Bakers
688	Food batchmakers
	<i>Precision Inspectors, Testers, and Related Workers</i>
689	Inspectors, testers, and graders
693	Adjusters and calibrators
	Plant and System Operators
694	Water and sewage treatment plant operators
695	Power plant operators
696	Stationary engineers
699	Miscellaneous plant and system operators
	OPERATORS, FABRICATORS, AND LABORERS
	Machine Operators, Assemblers, and Inspectors
	<i>Machine Operators and Tenders, except Precision Metal Working and Plastic Working</i>
	<i>Machine Operators</i>
703	Lathe and turning machine set-up operators
704	Lathe and turning machine operators
705	Milling and planing machine operators
706	Punching and stamping press machine operators
707	Rolling machine operators
708	Drilling and boring machine operators
709	Grinding, abrading, buffing, and polishing machine operators
713	Forging machine operators
714	Numerical control machine operators
715	Miscellaneous metal, plastic, stone, and glass working machine operators
717	Fabricating machine operators, n.e.c.
	<i>Metal and Plastic Processing Machine Operators</i>
719	Molding and canting machine operators
723	Metal plating machine operators
724	Heat treating equipment operators
725	Miscellaneous metal and plastic processing machine operators

1990 Census Code	Industry
	<i>Woodworking Machine Operators</i>
726	Wood lathe, routing, and planing machine operators
727	Sawing machine operators
728	Shaping and joining machine operators
729	Nailing and tacking machine operators
733	Miscellaneous woodworking machine operators
	<i>Printing Machine Operators</i>
734	Printing press operators
735	Photoengravers and lithographers
736	Typesetters and compositors
737	Miscellaneous printing machine operators
	<i>Textile, Apparel, and Furnishings Machine Operators</i>
738	Winding and twisting machine operators
739	Knitting, looping, taping, and weaving machine operators
743	Textile cutting machine operators
744	Textile serving machine operators
745	Shoe machine operators
747	Pressing machine operators
748	Laundering and dry cleaning machine operators
749	Miscellaneous textile machine operators
	<i>Machine Operators, Assorted Materials</i>
753	Cementing and gluing machine operators
754	Packaging and filling machine operators
755	Extruding and forming machine operators
756	Mixing and blending machine operators
757	Separating, filtering, and clarifying machine operators
758	Compressing and compacting machine operators
759	Painting and paint spraying machine operators
763	Roasting and baking machine operators, food
764	Washing, cleaning, and pickling machine operators
765	Folding machine operators
766	Furnace, kiln, and oven operators, exc. food
768	Crushing and grinding machine operators
769	Slicing and cutting machine operators
773	Motion picture projectionists
774	Photographic process machine operators
777	Miscellaneous machine operators, n.e.c.
779	Machine operators, not specified
	<i>Fabricators, Assemblers, and Hand Working Occupations</i>
783	Welders and cutters
784	Solderers and brazers
785	Assemblers
786	Hand cutting and trimming occupations
787	Hand molding, casting, and forming occupations
789	Hand painting, coating, and decorating occupations
793	Hand engraving and printing occupations
795	Miscellaneous hand working occupations

1990 Census Code	Industry
	<i>Production Inspectors, Testers, Samplers, and Weighers</i>
796	Production inspectors, checkers, and examiners
797	Production testers
798	Production samplers and weighers
799	Graders and sorters, exc. agricultural
	Transportation and Material Moving Occupations
	<i>Motor Vehicle Operators</i>
803	Supervisors, motor vehicle operators
804	Truck drivers
806	Driver-sales workers
808	Bus drivers
809	Taxicab drivers and chauffeurs
813	Parking lot attendants
814	Motor transportation occupations, n.e.c.
	<i>Transportation Occupations, Except Motor Vehicles</i>
	<i>Rail Transportation Occupations</i>
823	Railroad conductors and yardmasters
824	Locomotive operating occupations
825	Railroad brake, signal, and switch operators
826	Rail vehicle operators, n.e.c.
	<i>Water Transportation Occupations</i>
828	Ship captains and mates, except fishing boats
829	Sailors and deckhands
833	Marine engineers
834	Bridge, lock, and lighthouse tenders
	<i>Material Moving Equipment Operators</i>
843	Supervisors, material moving equipment operators
844	Operating engineers
845	Longshore equipment operators
848	Hoist and winch operators
849	Crane and tower operators
853	Excavating and loading machine operators
855	Grader, dozer, and scraper operators
856	Industrial truck and tractor equipment operators
859	Miscellaneous material moving equipment operators
	Handlers, Equipment Cleaners, Helpers, and Laborers
864	Supervisors, handlers, equipment cleaners, and laborers, n.e.c.
865	Helpers, mechanics and repairers
	<i>Helpers, Construction and Extractive Occupations</i>
866	Helpers, construction trades
867	Helpers, surveyor
868	Helpers, extractive occupations
869	Construction laborers
874	Production helpers

1990 Census Code	Industry
	<i>Freight, Stock, and Material Handlers</i>
875	Garbage collectors
876	Stevedores
877	Stock handlers and baggers
878	Machine feeders and offbearers
883	Freight, stock, and material handlers, n.e.c.
885	Garage and service station related occupations
887	Vehicle washers and equipment cleaners
888	Hand packers and packagers
889	Laborers, except construction
	MILITARY OCCUPATIONS
905	Military Occupation

Introduction to the Created Variable Appendices

The mainfile NLSY97 CD-ROM contains a number of created variables. These created variables will be included on each public release of the NLSY97. With very few exceptions, these variables are either commonly used items that are derived from a number of different NLSY97 survey questions or longitudinal items that require updating in each round. If they were not provided, creating either the longitudinal or the cross-sectional variables might present difficulties for some NLSY97 users. In general, the created variables present information in five topical areas: education, employment, family background, geographic information, and income and assets.

The first five appendices present the programs that were used to create the NLSY97 variables in round 1. The majority of programs are in SAS; a few are in SPSS. In the interest of space, six variables were not included in these appendices. These are the following:

CV_INTERVIEW_CMONT
CV_INTERVIEW_DATE
CV_AGE_INT_DATE
CV_AGE(MONTHS)_INT_DATE
CV_PARENT_INT_DATE
CV_AGE_12/31/96

These variables are either directly picked up from the data set or relatively easy to compute. In the case of CV_INTERVIEW_CMONT and CV_AGE(MONTHS)_INT_DATE, a simple formula of {[variable year-1980]*12}+variable month was used to translate the date into a continuous month scheme (see Appendix 7 for a further explanation of the continuous month scheme).

These programs are provided for the convenience of the user and are available electronically from NLS User Services (see the Table of Contents for information on how to contact User Services).

Appendices 6 and 7 provide detailed information about the creation of the event history variables, a special set of variables included on the mainfile CD-ROM. Although programs are not included, the text of Appendix 6 describes the various status arrays available and discusses the creation and editing process. Appendix 7 explains the continuous month and continuous week systems used in the creation of event history variables and provides a crosswalk of continuous month/week and actual dates so that researchers can associate event history variables with other information about the respondents.

INDEX TO CREATED VARIABLE PROGRAMS

This section lists by question name each NLSY97 created variable for which a program is provided in this document, along with the accompanying page number for the program.

CV_AMT_GOVNT_PGM_PCY.xx, 173	CV_HH_REL_AGE_12, 133
CV_ASSOC_CREDITS, 38	CV_HH_REL_AGE_2, 133
CV_BA_CREDITS, 38	CV_HH_REL_AGE_6, 133
CV_BIO_CHILD_HH, 113	CV_HH_REL_BIRTH, 108
CV_BIO_CHILD_NR, 113	CV_HH_SIZE, 103
CV_BIO_MOM AGE CHILD1, 121	CV_HH_UNDER_18, 103
CV_BIO_MOM AGE YOUTH, 121	CV_HH_UNDER_6, 103
CV_CENSUS_REGION, 141	CV_HIGHEST_DEGREE_EVER, 36
CV_CENSUS_REGION AGE 12, 141	CV_HOURS_WK_EVER, 92
CV_CHILD_BIRTH_DATE.xx_M, 113	CV_HOURS_WK_YR.xx, 92
CV_CHILD_BIRTH_DATE.xx_Y, 113	CV_HRLY_COMPENSATION, 74
CV_CHILD_BIRTH_MONTH.xx, 113	CV_HRLY_PAY, 62
CV_CHILD_DEATH_DATE.xx_M, 113	CV_HS_DIPLOMA, 32
CV_CHILD_DEATH_DATE.xx_Y, 113	CV_INCOME_GROSS_YR, 145
CV_CHILD_DEATH_MONTH.xx, 113	CV_JOB<13_WKS, 74
CV_CHILD_STATUS.xx, 113	CV_MARRIAGES_TTL, 118
CV_CITIZENSHIP, 128	CV_MARSTAT, 106
CV_COHAB_TTL, 118	CV_MARSTAT_COLLAPSED, 106
CV_ENROLLSTAT, 29	CV_MSA, 141
CV_ESR, 59	CV_MSA AGE 12, 141
CV_ESR_COLLAPSED, 59	CV_PIAT_PERCENTILE_SCORE, 39
CV_FIRST_COHAB_DATE_M, 118	CV_PIAT_STANDARD_SCORE, 39
CV_FIRST_COHAB_DATE_Y, 118	CV_RESIDENCES_TTL, 37
CV_FIRST_COHAB_MONTH, 118	CV_SCH_ATTEND_YR, 37
CV_FIRST_MARRY_DATE_M, 118	CV SCHOOL_SIZE, 44
CV_FIRST_MARRY_DATE_Y, 118	CV SCHOOL_TYPE, 31
CV_FIRST_MARRY_MONTH, 118	CV_STUDENT_TEACHER_RATIO, 44
CV_GED, 32	CV_TTL_JOBS_EVER, 99
CV_GOVNT_PGM_EVER, 173	CV_TTL_JOBS_YR.xx, 97
CV_GOVNT_PGM_YR.xx, 173	CV_URBAN_RURAL, 142
CV_GRADE_SKIPPED_EVER, 33	CV_URBAN_RURAL AGE 12, 142
CV_GRADES_REPEAT_EVER, 33	CV_WKSWK_DLI, 86
CV_HGC_EVĒR, 35	CV_WKSWK EVER, 86
CV_HH_INCOME_SOURCE, 145	CV_WKSWK_JOB_DLI.xx, 91
CV_HH_NET_WORTH_P, 145	CV_WKSWK_JOB_YR.xx.xx, 89
CV_HH_NET_WORTH_Y, 145	CV_WKSWK_YR.xx, 86
CV_HH_POV_RATIO, 145	

NLSY97 Appendix 1:

Education Variable Creation

YOUTH'S ENROLLMENT STATUS AS OF THE SURVEY DATE

Variables Created: CV_ENROLLSTAT

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
YS600	YSCH-600	YS4000	YSCH-4000
YS610	YSCH-610	YS15901	YSCH-15900.01
YS620	YSCH-620	YS11700	YSCH-11700
YS670	YSCH-670	YS26700	YSCH-26700
YS1600	YSCH-1600	YS13300	YSCH-13300
YS1800	YSCH-1800	YS28200	YSCH-28200
YS3500	YSCH-3500	PUBID	PUBID

Codes for Created Variable

1 = not enrolled, no dipl/GED	7 = not enrolled, graduate degree
2 = not enrolled, GED	8 = enrolled, grades 1-12
3 = not enrolled, HS Diploma	9 = enrolled, 2-year college
4 = not enrolled, some college	10 = enrolled, 4-year college
5 = not enrolled, 2 year college graduate	11 = enrolled, graduate program
6 = not enrolled, 4 year college graduate	

This program creates an enrollment status variable for each respondent.

```
***** PRELIMINARIES *****
encat = -4; /* Initialize */

*** CLEAR 'UNGRADED' RESPONSES;
if YS3500=95 then do;
  YS3500=.;
end;

*** CREATE NO DIPLOMA/GED INDICATOR;
if YS11700<1 and YS26700<1 and YS13300<1 and YS28200<1
then nodip=1;

*** CREATE HIGHEST GRADE EVER ATTENDED MEASURE;
HGA=max(YS3500,YS4000);

***** DETERMINE ENROLLMENT STATUS *****
enrolled = 1;
/* If not enrolled and not in summer school and not on summer break */
if (YS600=0 and YS610<1 and YS670<1)
then enrolled = 0;

/* If not enrolled or don't know and gave a reason for leaving school */
if (-2<=YS600<=0 and YS1600>0)
  then enrolled = 0;

*** NOT CURRENTLY ENROLLED ***;
if (enrolled = 0) then do;
  if nodip=1      /* No Diploma */
    then encat=1;

  if YS11700=1 or YS26700=1 /* Got a HS diploma */
    then encat=3;
```

```
/* Didn't get a HS Diploma and did get a GED */
if (YS11700 ne 1 and YS26700 ne 1) and (YS13300=1 or YS28200=1)
then encat=2;

/* Last school attended was some type of college */
if (YS1800=4 or YS1800=5)
then encat=4;
end;

*** CURRENTLY ENROLLED ***;
if (enrolled = 1) then do;

/* Attending grades 1 to 12 or attending college, don't know and HGA not college and no diploma */
if 1<=YS1800<=3 or ((YS1800>3 or YS1800=-2) and HGA<13 and nodip=1) then do;
    encat=8;
end;

/* Enrolled in 2 year college and current grade beyond High School */
if YS1800=4 and (95>max(YS3500,YS4000)>=13)
then encat=9;

/* Enrolled in 4 year college and current grade beyond High School */
if YS1800=5 and (95>max(YS3500,YS4000)>=13) then do;
    /* Working toward MA, Ph.d. or professional degree */
    if 4<=YS15901<=6 then encat=7;
    else encat=10;
end;

end;

***HAND EDITS;
***Listed "Summer Vacation" as reason for non-enrollment (YS1600=other);
if pubid=4585 then encat=4;
if pubid=4595 then encat=4;
if pubid=4621 then encat=4;
if pubid=8191 then encat=4;

***Listed in a business or nursing program;
if encat=., then do;
    nocat=1;
end;

endsas;
```

CURRENT OR MOST RECENT SCHOOL PRIVATE OR PAROCHIAL

Variables Created: CV_SCHOOL_TYPE

Variables Used

Name in Program	Question Name on CD
YS1800	YSCH-1800
YS3400	YSCH-3400

Codes for Created Variable

- 1 = Public school
- 2 = Parochial
- 3 = Private (non-parochial)
- 4 = Other
- 0 = Missing (Last school not grade 1-12 or description ("kind") missing)

This program identifies the type of school attended by the respondent. Users should note that school type is defined only for respondents attending grades 1-12.

```
schlcat=0;
if 1<=YS1800<=3 then do;
  if YS3400=1 then schlcat=1;
  else if 3<=YS3400<=4 then schlcat=2;
  else if YS3400=5 then schlcat=3;
  else if (YS3400=2 or YS3400=6 or YS3400=9) then schlcat=4;
  else if YS3400=-1 then schlcat=-1;
  else if YS3400=-2 then schlcat=-2;
end;

/* Type of school cannot be determined for respondents who say they are currently enrolled in college */

if YS1800>3 then do;
  schlcat=0;
end;

endsas;
```

Note:

There are 6 individuals who are not currently enrolled who provide no information for YS1800 and YS3400. Schlcat is set equal to missing for these respondents.

DATE RECEIVED DIPLOMA OR DEGREE

Variables Created: CV_GED
CV_HS_DIPLOMA

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
YS11700	YSCH-11700	YS26700	YSCH-26700
YS11900_M, _Y	YSCH-11900_M, _Y	YS26800_M, _Y	YSCH-26800_M, _Y
YS13300	YSCH-13300	YS28200	YSCH-28200
YS13500_M, _Y	YSCH-13500_M, _Y	YS28300_M, _Y	YSCH-28300_M, _Y

This program creates two continuous month variables, one identifying the month that the respondent received a GED and the other identifying the month that the respondent received a high school diploma. In future rounds, similar variables will calculate the month the respondent received an associate's, bachelor's, master's, doctoral, or professional degree; no respondents had received these degrees in round 1. For more information on the continuous month scheme, see appendix 7 in this document.

```

HSdate = -4;           /* Initialize */
GEDdate = -4;

/* Check if the respondent graduated from High School */

if (YS11700=1) and (YS119_Y > 0) and (YS119_M > 0)
then HSdate=((YS119_Y)-1980)*12 + YS119_M;
if (YS11700=1) and (YS119_Y > 0) and (YS119_M < 0)
then HSdate=((YS119_Y)-1980)*12 + 6; /* Missing defaults to June */

if (YS26700=1) and (YS268_Y > 0) and (YS268_M > 0)
then HSdate=((YS268_Y)-1980)*12 + YS268_M;
if (YS26700=1) and (YS268_Y > 0) and (YS268_M < 0)
then HSdate=((YS268_Y)-1980)*12 + 6; /* Missing defaults to June */

/* Check if the respondent received a GED */

if (YS13300=1) and (YS135_Y > 0) and (YS135_M > 0)
then GEDdate=((YS135_Y)-1980)*12 + YS135_M;
if (YS13300=1) and (YS135_Y > 0) and (YS135_M < 0)
then GEDdate=((YS135_Y)-1980)*12 + 6; /* Missing defaults to June */

if (YS28200=1) and (YS283_Y > 0) and (YS283_M > 0)
then GEDdate=((YS283_Y)-1980)*12 + YS283_M;
if (YS28200=1) and (YS283_Y > 0) and (YS283_M < 0)
then GEDdate=((YS283_Y)-1980)*12 + 6; /* Missing defaults to June */

endsas;

```

NUMBER OF GRADES REPEATED OR SKIPPED

Variables Created: CV_GRADES_REPEAT_EVER
CV_GRADE_SKIPPED_EVER

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
PINF015Y	PINF-015_Y	PC814601-02	PC8-146.01-.02
PC814101	PC8-141	PC814701-01	PC8-147.01-.02
PC814201-13	PC8-142_001-_013	PC814801-02	PC8-148.01-.02
PC814301	PC8-143		

This program creates variables counting the number of grades the respondent ever repeated or skipped based on information from the parent interview. If there was no parent interviewed, the variables are coded as valid skips (-4).

*****GRADES REPEATED SECTION*****

*CALCULATE THE TOTAL GRADES REPEATED AS OF INTERVIEW DATE

*<INITIALIZE EVERYONE TO ZERO>
COMPUTE GRRPET = 0.
IF (PINF015Y=-4) GRRPET = -4.
*<If there is a valid information on grades repeated, use it.

IF (PC814101<0) GRRPET=PC814101.
DO IF (PC814101>0) AND (GRRPET=0).
COUNT GRRPET= PC814201 PC814202 PC814203 PC814204 PC814205 PC814206 PC814207
PC814208 PC814209 PC814210 PC814211 PC814212 PC814213 (1 THRU HIGHEST).
END IF.

*****GRADES SKIPPED SECTION*****

*CALCULATE THE TOTAL GRADES SKIPPED AS OF INTERVIEW DATE

*<INITIALIZE EVERYONE TO ZERO>
COMPUTE GRSKIP= 0.
IF (PINF015Y=-4) GRSKIP = -4.
IF (PC814301<0) GRSKIP = PC814301.

* <SET KINDERGARTEN RESPONSES TO ZERO in order to compute the number of grades skipped, as the raw data recorded pre-K and Kindergarten as 90, 91, or 92>

DO IF (PC814601>=90).
COMPUTE PC814601=0.
END IF.
DO IF (PC814602>=90).
COMPUTE PC814602=0.
END IF.

*<If there is a valid information on grades skipped, use it.>

IF (PC814601=-1 OR PC814701=-1 OR PC814602=-1 OR PC814702=-1) GRSKIP=-1.
IF (PC814601=-2 OR PC814701=-2 OR PC814602=-2 OR PC814702=-2) GRSKIP=-2.
DO IF (PC814301=1 AND GRSKIP=0).
COMPUTE SKIP1= (PC814701-PC814601).
COMPUTE SKIP2= (PC814702-PC814602).

- * <If a skip is indicated, but the result is invalid (i.e., negative number) OR if two skips are indicated but the start of the second skip a lower grade than indicated on the first skip (i.e., the first loop indicated the youth skipped the second grade but the second loop shows the student in the second grade), THEN the skip will be coded as an invalid skip.>

IF (PC814701<PC814601 AND PC814601>0) SKIP1=-3.
IF (PC814602>PC814702 AND PC814602>0) SKIP2=-3.
IF ((PC814602<PC814601 OR PC814602>PC814701) AND PC814602>0) SKIP2=-3.

- * <As noted above, grades skipped are calculated with the assumption of an end-of-year skip. For example, the youth successfully completes the second (2) grade at the end of the school year, but begins the fourth (4) grade at the beginning of the following academic year. In this case, the calculation of this raw data into SKIP1 (or SKIP2) would erroneously indicate a 2-grade skip. To account for this, results indicating skipping more than one grade are decreased by one to more accurately portray the number of grades skipped.>

```
IF (SKIP1>1) SKIP1=SKIP1 - 1.  
IF (SKIP2>1) SKIP2=SKIP2 - 1.  
END IF.  
DO IF (SKIP1=0 OR SKIP2=0).  
COMPUTE CHECK=1.  
ELSE.  
COMPUTE CHECK=0.  
END IF.  
DO IF (SKIP1>=0 AND SKIP2>=0).  
COMPUTE GRSKIP=SKIP1+SKIP2.  
ELSE IF (SKIP1>0 AND SKIP2=-3).  
COMPUTE GRSKIP=SKIP1.  
ELSE IF (SKIP1=-3 AND SKIP2>0).  
COMPUTE GRSKIP=SKIP2.  
END IF.  
  
EXECUTE.
```

HIGHEST GRADE COMPLETED AS OF THE SURVEY DATE

Variables Created: CV_HGC_EVER

Variables Used

Name in Program	Question Name on CD
YS5000	YSCH-5000

Codes for Created Variable

- 1-20 = actual grade completed
- 0 = valid skip/missing/none
- 95 = ungraded

This variable is essentially a straight pick-up with some minor modifications. Individuals who skip this portion of the survey because they are never enrolled (e.g. if home schooled) are put in category 0. These are valid skips (-4) in the raw data. A respondent currently enrolled in (but not having completed) grade 1 will also report a highest grade completed = 0.

```
HGC=0;  
if -3<=YS5000<=95 then HGC=YS5000;  
endsas;
```

HIGHEST DEGREE RECEIVED AS OF THE SURVEY DATE

Variables Created: CV_HIGHEST_DEGREE_EVER

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
YS600	YSCH-600	YS4000	YSCH-4000
YS610	YSCH-610	YS15901	YSCH-15900.01
YS620	YSCH-620	YS11700	YSCH-11700
YS670	YSCH-670	YS26700	YSCH-26700
YS1600	YSCH-1600	YS13300	YSCH-13300
YS1800	YSCH-1800	YS28200	YSCH-28200
YS3500	YSCH-3500		

Codes for Created Variable

0 = none	4 = Bachelors
1 = GED	5 = Masters
2 = HS Diploma	6 = Doctoral
3 = Associates	7 = Professional

This program identifies the highest degree or diploma received by the respondent. In round 1, no respondent had completed a degree higher than a high school diploma and/or GED. Therefore, in practice, the range of actual values is limited to [0,2].

*** Initialize Everyone to NO DIPLOMA/GED INDICATOR;

degree=0;

*** CREATE GED INDICATOR;

if YS13300=1 or YS28200=1 then do;
 degree=1;
end;

*** CREATE HS DIPLOMA INDICATOR;

if YS11700=1 or YS26700=1 then do;
 degree=2;
end;

endsas;

NUMBER OF SCHOOLS ATTENDED, NUMBER OF RESIDENCES

Variables Created: CV_SCH_ATTEND_YR
CV_RESIDENCES_TTL

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
PINF015Y	PINF-015_Y	PC8043	PC8-043
PC8097	PC8-097	PUBID	PUBID
PC812801-09	PC8-128.01-09		

This program calculates the number of schools ever attended by the respondent and the number of residences in which youth has ever lived. Both of these variables are created with information from the parent interview. If no parent was interviewed (as indicated by PINF015Y), the created variables are coded as valid skips (-4).

The schools attended variable is calculated by adding up all the valid responses on each loop of information. If the youth has never attended school, as indicated by PC8097, then SCHLSATT is coded as zero (no schools attended). If the initial question (PC8-128) has a response of don't know or refused (-1 or -2), then SCHLSATT is coded as such. Number of different residences lived since 12th birthday utilizes the data from PC8043. Valid skip, refused, and don't know responses are coded as such in the created variable.

*****SCHOOLS ATTENDED SECTION*****

*<Initialize everyone to zero>
COMPUTE SCHLSATT = 0.
IF (PC812801=-4) SCHLSATT = -4.

*<If there is a valid information on schools attended, use it.
DO IF (SCHLSATT<=0).
COUNT NUMSCHLS= PC812801 PC812902 PC812903 PC812904 PC812905 PC812906 PC812907
PC812908 PC812909 (0 THRU HIGHEST).
COMPUTE SCHLSATT=NUMSCHLS.
IF (PC8097=99 AND NUMSCHLS<1) SCHLSATT=0.
END IF.
IF (PINF015Y=-4) SCHLSATT = -4.
IF (PC812801=-1 OR PC812801=-2) SCHLSATT=PC812801.

*****NUMBER OF DIFFERENT RESIDENCES LIVED SINCE YOUTH'S 12TH BIRTHDAY SECTION****

*Calculate the number of different residences lived since youth's 12th birthday as of interview date

*<Initialize everyone to RSLIVED=1>
COMPUTE RSLIVED=1.
IF (PINF015Y=-4) RSLIVED = -4.
IF (PC8043=-4) RSLIVED = -4.

*<If there is a valid information on residences lived, use it.
DO IF (PC8043>=0).
COMPUTE RSLIVED=RSLIVED+PC8043.
ELSE IF (PC8043<0) AND (PC8043>-4).
COMPUTE RSLIVED=PC8043.
END IF.

EXECUTE.

TOTAL FRACTION OF CREDITS EARNED TOWARDS BACHELORS/ASSOCIATE DEGREE

Variables Created: CV_BA_CREDITS
CV_ASSOC_CREDITS

Variables Used

Name in Program	Question Name on CD
PUBID	PUBID
YS1800	YSCH-1800
totcr	YSCH-1600.01
earnrcr	YSCH-22800.01

Codes for Created Variable

These variables are created on a continuous scale. They have 2 implied decimal places.

This program calculates the fraction of credits the respondent has completed toward an associate's degree or a bachelor's degree. YS1800 (Type of School) is used to determine the type of degree sought because degree type is not specifically asked in the survey. The default value is set to -4. This variable is only created for those possibly enrolled in some form of college (YS1800>=4) regardless of created enrollment status. For individuals currently enrolled in a college who have not yet started taking their first classes, questions YS16001 and YS22801 have valid skips. Therefore, the default in this case is set to 0.

```
*** INITIALIZE VARIABLES;
bachfrac=-4;
assfrac=-4;

/* Compute fraction for respondents attending a 2 year college */
if YS1800=4 then do;
    assfrac=0;          /* Initialize to zero for those enrolled */
    if totcr=-1 or earnrcr<=-1 then do;
        assfrac=-1; end;
    if totcr=-2 or earnrcr=-2 then do;
        assfrac=-2; end;
    if totcr=-3 or earnrcr=-3 then do;
        assfrac=-3; end;
    if totcr>0 and earnrcr>0 then do;
        assfrac=(earnrcr/totcr)*100; end;
    end;
assfrac=round(assfrac,1);

/* Compute fraction for respondents attending a 4 year college */
if YS1800=5 then do;
    bachfrac=0;          /* Initialize to zero for those enrolled */
    if totcr=-1 or earnrcr=-1 then do;
        bachfrac=-1; end;
    if totcr=-2 or earnrcr=-2 then do;
        bachfrac=-2; end;
    if totcr>0 and earnrcr>0 then do;
        bachfrac=(earnrcr/totcr)*100; end;
    end;
bachfrac=round(bachfrac,1);

/* HAND EDITS Both individuals are finishing high school dipl./GED at a 2-year college */
if pubid=5901 or pubid=7652 then assfrac=-4;

endsas;
```

YOUTH'S MATH PIAT SCORE

Variables Created: CV_PIAT_STANDARD_SCORE
CV_PIAT_PERCENTILE_SCORE

Variables Used

Name in Program	Question Name on CD
I900D, M, Y	YINF-900_D, _M, _Y
DOB01D, M, Y	KEY!BDATE_D, _M, _Y
PIATRAW	PIAT_RAW_SCORE_REVISED

This program calculates the respondent's age utilizing the interview date information and the respondent's date of birth information. It then uses the age data and the raw PIAT score to create a standard PIAT score and percentile rank for each respondent.

*****INTERVIEW DATE SECTION (CONTINUOUS MONTHS FORMAT)*****

*<INITIALIZE EVERYONE TO VALID SKIP>

```
COMPUTE INTDAY = -4.  
COMPUTE INTMONTH = -4.  
COMPUTE INTYEAR = -4.
```

```
DO IF (I900M >=0) AND (I900D >=0).  
    COMPUTE INTDAY=I900D.  
    COMPUTE INTMONTH=I900M.  
    COMPUTE INTYEAR=I900Y.  
    COMPUTE CONTMINT=((INTYEAR-1980)*12)+INTMONTH.  
END IF.
```

*****RESPONDENT AGE SECTION*****

*<If there is a valid date of birth for respondent, use it.>

```
COMPUTE DOBDAY = DOB01D.  
COMPUTE DOBMONTH = DOB01M.  
COMPUTE DOBYEAR = DOB01Y.
```

*<Ensure all respondents have an in range value.>

```
IF (DOBDAY < 0) DOBDAY = 15.  
IF (DOBMONTH < 0) DOBMONTH = 6.  
IF (DOBYEAR < 1970) DOBYEAR = 1983.
```

*<CALCULATE THE YOUTH'S AGE IN CONTINUOUS MONTHS>

```
COMPUTE YAGEMNTH=((INTYEAR - DOBYEAR)*12)+(INTMONTH - DOBMONTH).  
IF (INTDAY-DOBDAY<0) YAGEMNTH=YAGEMNTH-1.
```

/* At this point the program loops through each three-month age range and assigns appropriate standard scores based on the respondent's raw score. This code is quite lengthy and is not reproduced here; instead, we include a table that provides the standard score and percentile rank associated with each raw score in each age range. Users who need the precise programming code should contact NLS User Services. */

Appendix 1: Education Variable Creation

Standard Score	Percentile Rank	Raw Score by Age Range (Years-Months)												
		12-0 to 12-2	12-3 to 12-5	12-6 to 12-8	12-9 to 12-11	13-0 to 13-2	13-3 to 13-5	13-6 to 13-8	13-9 to 13-11	14-0 to 14-2	14-3 to 14-5	14-6 to 14-8	14-9 to 14-11	
55	0	≤27	≤28	≤29	≤29	≤30	≤31	≤32	≤33	≤34	≤35	≤36	≤37	
56	0				30	31	32	33	34	35	36	37	38	
57	0	28	29	30	31	32	33	34	35	36	37	38	39	
58	0	29	30	31	32	33	34	35	36	37	38		40	
59	0	30	31	32	33	34	35	36	37			39		
60	0	31	32	33	34	35	36			38	39	40	41	
61	0	32	33	34	35	36	37	38	39	40	41	42		
62	1	33	34	35	36	37		38	39	40	41	42	43	
63	1	34	35	36	37		38	39	40	41	42	43	44	
64	1	35	36	37	38	38	39	40	41	42	43	44	45	
65	1		37	38		39	40	41	42	43	44	45	46	
66	1	36			39	40	41	42	43	44	45	46	47	
67	1	37	38	39	40	41	42	43	44	45	46	47	48	
68	2	38	39	40	41	42	43	44	45	46	47	48	49	
69	2	39	40	41	42	43	44	45	46	47	48	49	50	
70	2	40	41	42	43	44	45	46	47	48	49	50	51	
71	3	41	42	43	44	45	46	47	48	49	50	51	52	
72	3	42	43	44	45	46	47	48	49	50	51	52	53	
73	4	43	44	45	46	47	48	49	50	51	52	53	54	
74	4	44	45	46	47	48	49	50	51	52	53	54	55	
75	5		46	47	48	49	50	51	52	53	54	55	56	
76	5	45	47	48	49	50	51	52	53	54	55	56	57	
77	6	46		49	50		52		54		56			
78	7	47	48	50	51	51	53	53	55	55	57	57	58	
79	8	48	49			52		54		56		58	59	
80	9	49	50	51	52	53	54	55	56	57	58	59	60	
81	10	50	51	52	53	54	55	56	57	58	59	60	61	
82	12	51	52	53	54	55	56	57	58	59	60	61	62	
83	13		53	54	55	56	57	58	59	60	61	62	63	
84	14	52	54	55	56	57	58	59	60	61	62	63	64	
85	16	53		56	57	58	59	60	61	62	63	64	65	
86	18	54	55	57	58	59	60	61	62	63	64	65	66	
87	19	55	56								65			
88	21	56	57	58	59	60	61	62	63	64		66	67	
89	23		58	59	60	61	62	63	64	65	66	67	68	
90	25	57		60	61	62	63	64	65	66	67	68	69	
91	27	58	59					65	66	67	68	69	70	
92	30	59	60	61	62	63	64				69			
93	32			62	63	64	65	66	67	68		70	71	
94	34	60	61		64	65	66	67	68	69	70	71	72	
95	37	61	62	63				68	69	70	71	72	73	
96	39		63	64	65	66	67							
97	42	62			66	67	68	69	70	71	72	73	74	
98	45	63	64	65			69	70	71	72	73	74	75	
99	47		65	66	67	68								
100	50	64		67	68	69	70	71	72	73	74	75	76	

Appendix 1: Education Variable Creation

Standard Score	Percentile Rank	Raw Score by Age Range (Years-Months)												
		12-0 to 12-2	12-3 to 12-5	12-6 to 12-8	12-9 to 12-11	13-0 to 13-2	13-3 to 13-5	13-6 to 13-8	13-9 to 13-11	14-0 to 14-2	14-3 to 14-5	14-6 to 14-8	14-9 to 14-11	
101	53	65	66				71	72	73	74	75	76	77	
102	55		67	68	69	70	72	73	74	75	76	77	78	
103	58	66	68	69	70	71			75			78		
104	61	67		70	71	72	73	74		76	77		79	
105	63	68	69		72	73	74	75	76	77	78	79	80	
106	66	69	70	71		74	75	76	77	78	79	80	81	
107	68		71	72	73		76	77	78	79	80	81	82	
108	70	70	72	73	74	75	77	78	79	80	81	82	83	
109	73	71		74	75	76				81	82	83		
110	75	72	73	75	76	77	78	79	80	82	83	84	84	
111	77	73	74	76	77	78	79	80	81				85	
112	79	74	75		78	79	80	81	82	83	84	85	86	
113	81	75	76	77	79	80	81	82	83	84	85	86	87	
114	82	76	77	78	80	81	82	83	84	85	86	87	88	
115	84		78	79		82	83	84	85	86	87	88	89	
116	86	77	79	80	81	83	84	85	86	87	88	89	90	
117	87	78	80	81	82	84	85	86	87	88	89	90	91	
118	88	79		82	83		86	87	88	89	90	91	92	
119	90	80	81	83	84	85	87	88	89	90	91	92	93	
120	91	81	82	84	85	86	88	89	90	91	92	93	94	
121	92		83		86	87		90	91	92	93	94	95	
122	93	82	84	85	87	88	89	91	92	93	94	95	96	
123	94	83	85	86		89	90							
124	95	84	86	87	88	90	91	92	93	94	95	96	97	
125	95	85		88	89		92	93	94	95	96	97	98	
126	96	86	87	89	90	91	93	94	95	96	97	98	99	
127	96		88		91	92		95	96	97	98	99	100	
128	97	87	89	90	92	93	94	96	97	98	99	100		
129	97	88	90	91		94	95							
130	98	89	91	92	93	95	96	97	98	99	100			
131	98	90		93	94		97	98	99	100				
132	98		92		95	96	98	99	100					
133	99	91	93	94	96	97		100						
134	99	92	94	95	97	98	99							
135	99	93		96		99	100							
136	99	94	95	97	98	100								
137	99		96	98	99									
138	99	95	97		100									
139	100	96	98	99										
140	100	97		100										
141	100	98	99											
142	100		100											
143	100	99												
144	100	100												

Appendix 1: Education Variable Creation

Standard Score	Percentile Rank	Raw Score by Age Range (Years-Months)												
		15-0 to 15-2	15-3 to 15-5	15-6 to 15-8	15-9 to 15-11	16-0 to 16-2	16-3 to 16-5	16-6 to 16-8	16-9 to 16-11	17-0 to 17-2	17-3 to 17-5	17-6 to 17-8	17-9 to 17-11	
55	0	≤38	≤39	≤40	≤41	≤42	≤42	≤43	≤43	≤44	≤44	≤44	≤45	
56	0	39	40	41	42		43	44	44	45	45	45	46	
57	0	40	41	42		43	44	45	45		46	46	47	
58	0	41	42	43	43	44	45	46	46	46	47	47	48	
59	0		43		44	45	46	47	47	47	48	48		
60	0	42		44	45	46	47		48	48		49	49	
61	0	43	44	45	46	47	48	48	49	49	49	50	50	
62	1	44	45	46	47	48	49	49	50	50	50		51	
63	1	45	46	47	48	49	50	50	51	51	51	51	52	
64	1	46	47	48	49	50	51	51	52	52	52	52		
65	1	47	48	49	50	51		52		53	53	53	53	
66	1	48	49	50	51	52	52	53	53	54	54	54	54	
67	1	49	50	51	52	53	53	54	54	55	55	55	55	
68	2	50	51	52	53		54	55	55	56	56	56	56	
69	2	51	52	53	54	54	55	56	56	57	57	57	57	
70	2	52	53	54	55	55	56	57	57	58	58	58	58	
71	3	53	54	55	56	56	57	58	58		59	59	59	
72	3	54	55	56	57	57	58	59	59	59	60	60	60	
73	4	55	56	57		58	59	60	60	60	61	61	61	
74	4	56	57	58	58	59	60	61	61	61	62	62	62	
75	5	57	58		59	60	61		62	62	63	63	63	
76	5	58	59	59	60	61	62	62	63	63	64	64	64	
77	6			60	61	62	63	63	64	64	65	65	65	
78	7	59	60	61	62	63	64	64	65	65	66	66	66	
79	8	60	61	62	63	64	65	65	66	66		67	67	
80	9	61	62	63	64	65		66		67	67	68	68	
81	10	62	63	64	65		66	67	67	68	68			
82	12	63	64	65	66	66	67	68	68	69	69	69	69	
83	13	64	65	66		67	68	69	69		70	70	70	
84	14	65	66	67	67	68	69	70	70	70	71	71	71	
85	16	66	67		68	69	70		71	71	72	72	72	
86	18	67		68	69	70	71	71	72	72		73	73	
87	19		68	69	70	71		72		73	73	74	74	
88	21	68	69	70	71	72	72	73	73	74	74	75	75	
89	23	69	70	71	72		73	74	74	75	75			
90	25	70	71	72		73	74		75			76	76	
91	27				73	74		75		76	76	77	77	
92	30	71	72	73	74	75	75	76	76	77	77	78	78	
93	32	72	73	74	75		76	77	77	78	78		79	
94	34	73	74	75		76	77		78		79	79		
95	37	74	75		76	77		78	79	79	80	80	80	
96	39			76	77	78	78	79		80		81	81	
97	42	75	76	77			79		80		81		82	
98	45	76	77		78	79	80	80	81	81	82	82		
99	47				78	79	80		81		82		83	
100	50	77	78	79	80		81	82	82	83	83	84	84	

Appendix 1: Education Variable Creation

Standard Score	Percentile Rank	Raw Score by Age Range (Years-Months)											
		15-0 to 15-2	15-3 to 15-5	15-6 to 15-8	15-9 to 15-11	16-0 to 16-2	16-3 to 16-5	16-6 to 16-8	16-9 to 16-11	17-0 to 17-2	17-3 to 17-5	17-6 to 17-8	17-9 to 17-11
101	53	78	79	80		81	82		83		84		85
102	55	79	80		81	82		83	84	84	85	85	
103	58			81	82	83	83	84		85		86	86
104	61	80	81	82	83		84	85	85	86	86	87	87
105	63	81	82	83	84	84	85		86	87	87		88
106	66	82	83	84		85	86	86	87		88	88	89
107	68	83	84		85	86		87	88	88	89	89	90
108	70	84	85	85	86	87	87	88		89	90	90	
109	73			86	87		88	89	89	90		91	91
110	75	85	86	87	88	88	89	90	90	91	91	92	92
111	77	86	87	88	89	89	90	91	91	92	92	93	93
112	79	87	88	89	90	90	91	92	92	93	93	94	94
113	81	88	89	90	91	91	92		93	94	94	95	95
114	82	89	90	91	92	92	93	93	94	95	95	96	96
115	84	90	91	92	93	93	94	94	95		96	97	97
116	86	91	92	93	94	94	95	95	96	96	97	98	98
117	87	92	93	94		95	96	96	97	97	98		99
118	88	93	94	95	95	96		97	98	98	99	99	100
119	90	94	95	96	96	97	97	98	99	99	100	100	
120	91	95	96		97	98	98	99		100			
121	92	96			97	98		99	100	100			
122	93	97	97	98	99	99	100						
123	94			98	99		100						
124	95	98	99	100	100								
125	95	99	100										
126	96	100											

Note: This table of *PIAT* scores is based on the information provided by the testing company. Users interested in more information about the *PIAT-Math Assessment* may wish to consult the following reference:

Markwardt, Frederick C. Jr. *Peabody Individual Achievement Test-Revised*. Circle Pines, Minn.: American Guidance Service, Inc., 1989.

QED DATA: SCHOOL SIZE AND STUDENT-TEACHER RATIO

Variables Created: CV_SCHOOL_SIZE
CV_STUDENT_TEACHER_RATIO

Variables Used

Name in Program	Question Name on CD
TYPESCH	YSCH-1800
CONTROL	YSCH-3400
GRADE	YSCH-3500
PUBID	PUBID
REGION	CV_CENSUS_REGION

Codes for Created Variable

School Size	Student-teacher ratio
1 = <100	1 = <14
2 = 100-299	2 = 14-17
3 = 300-499	3 = 18-21
4 = 500-749	4 = 22+
5 = 750-999	
6 = 1000+	

This program merges school identification information from the NLSY97 data with data from the “National Education Database,” provided under copyright by Quality Education Data (QED), Inc. It then creates two variables that provide information about the respondent’s school.

```

if CONTROL=999 then CONTROL=9;
SCHSIZE=-4;
STUDTEAC=-4;

/* Create SCHSIZE AND STUDTEAC while checking that the grade reported in the QED coincides with that
being reported in the NLSY97 */

if GRADSPAN='2' then GRADSPAN='3';
if GRADSPAN='A' then GRADSPAN='2';
if GRADSPAN='B' or GRADSPAN='C' or GRADSPAN='D' or GRADSPAN='E' or GRADSPAN='*' then
GRADSPAN='0';

dummy1=0; dummy4=0; dummy5=0; dummy6=0; dummy7=0; dummy8=0; dummy9=0; dummy10=0;

if GRADE ge 1 and GRADE le 3 then do;
dummy1=1; dummy4=1; dummy5=1; dummy6=0; dummy7=0; dummy8=0; dummy9=0; dummy10=0;
end;

if GRADE ge 4 and GRADE le 6 then do;
dummy1=1; dummy4=1; dummy5=1; dummy6=1; dummy7=0; dummy8=0; dummy9=0; dummy10=0;
end;

if GRADE ge 7 and GRADE le 8 then do;
dummy1=1; dummy4=0; dummy5=1; dummy6=1; dummy7=1; dummy8=1; dummy9=0; dummy10=0;
end;

if GRADE eq 9 then do;
dummy1=1; dummy4=0; dummy5=0; dummy6=0; dummy7=1; dummy8=0; dummy9=1; dummy10=0;
end;

```

```
if GRADE ge 10 and GRADE le 12 then do;
dummy1=1; dummy4=0; dummy5=0; dummy6=0; dummy7=0; dummy8=1; dummy9=1; dummy10=1;
end;

if GRADSPAN='1' and dummy1=1 then correct=1;
if GRADSPAN='4' and dummy4=1 then correct=1;
if GRADSPAN='5' and dummy5=1 then correct=1;
if GRADSPAN='6' and dummy6=1 then correct=1;
if GRADSPAN='7' and dummy7=1 then correct=1;
if GRADSPAN='8' and dummy8=1 then correct=1;
if GRADSPAN='9' and dummy9=1 then correct=1;
if GRADSPAN='2' and dummy10=1 then correct=1;
else correct=0;

/* If the grades reported in the QED and the NLSY97 do not coincide, both created variables equal -3; otherwise,
create them according to the definition provided at the top of the program */

if correct=0 then SCHSIZE=-3;
if correct=0 then STUDTEAC=-3;

if STUDRANG=0 or STUDRANG=1 then SCHSIZE=1;
if STUDRANG=2 then SCHSIZE=2;
if STUDRANG=3 then SCHSIZE=3;
if STUDRANG=4 then SCHSIZE=4;
if STUDRANG=5 then SCHSIZE=5;
if STUDRANG=6 or STUDRANG=7 then SCHSIZE=6;
if STUDRANG=. then SCHSIZE=-3;

if STUDENT1 gt 0 and TEACHERS gt 0 then STUDTEA1=STUDENT1/TEACHERS;
if STUDTEA1 lt 14 then STUDTEAC=1;
if STUDTEA1 ge 14 and STUDTEA1 lt 18 then STUDTEAC=2;
if STUDTEA1 ge 18 and STUDTEA1 lt 22 then STUDTEAC=3;
if STUDTEA1 ge 22 then STUDTEAC=4;
if STUDTEA1 eq . then STUDTEAC=-3;

/* Verify that there are not five or fewer schools in each cell (determined by the region, type of school, control, school
size and the student-teacher ratio) to avoid any possibility of identifying a school. Assign -3 to the created variables
for schools that fall in cells of five or fewer schools. */

data three;
set two;
INDEX=REGION2||TYPESCH||CONTROL||SCHSIZE||STUDTEAC;
proc freq;
tables REGION2*TYPESCH*CONTROL*SCHSIZE*STUDTEAC / OUT=A;
proc freq data=A;
tables COUNT;
data A;
set A;
INDEX=REGION2||TYPESCH||CONTROL||SCHSIZE||STUDTEAC;

data four;
set four;
if COUNT le 5 then SCHSIZE=-3;
if COUNT le 5 then STUDTEAC=-3;
proc freq;
endsas;
```

NLSY97 Appendix 2:

Employment Variable Creation

INTRODUCTION

A number of the created employment variables use the same program(s) as input. The two programs in this section are referred to throughout the employment variables. Thus, for example, to create the “Weeks Worked at Employee Job #x during 19xx” variables, survey staff first run the program below titled “emp_begin.sas” and then run the program included in the weeks worked section of this appendix.

EMP_BEGIN.SAS

***** The first part of this program reads in raw start and stop dates for each job (max=7) and converts them into NLSY97 week numbers. Some start/stop DAYS and MONTHS have been imputed if missing. Imputed days are coded as '15' and months as '6'. *****

```

array startm (i) starmo1-starmo10;
array startd (i) stardy1-stardy10;
array starty (i) staryr1-staryr10;
array stopm (i) stopmo1-stopmo10;
array stopd (i) stopdy1-stopdy10;
array stopy (i) stopyr1-stopyr10;

array sttdays (i) sttday1-sttday10;
/* total days that year from startdate (to Jan 1) */
array stpdays (i) stpday1-stpday10;
/* total days that year from stopdate (to Jan 1) */
array startwk (i) starw1-starw10;
array stopwk (i) stopw1-stopw10;
array srflag (i) srflg1-srflg10;
array spflag (i) spflg1-spflg10;

/**Fill-in start/stop day for those missing ***
/* flag1 = impute start day (valid month)
flag2 = impute start month (valid day)
flag3 = impute start day and month
flag4 = impute stop day (valid month)
flag5 = impute stop month (valid day)
flag6 = impute stop day and month */

flag1=0; flag2=0; flag3=0;
do over starty;
if starty>0 then do;
  if startm>0 and startd<=0 then do;
    startd=15; flag1=1; srflag=1; end;
  if startm<=0 and startd>0 then do;
    startm=6; flag2=1; srflag=1; end;
  if startm<=0 and startd<=0 then do;
    startm=6; startd=15; flag3=1; srflag=1; end;
end;
end;

flag4=0; flag5=0; flag6=0;
do over stopy;
if stopy>0 then do;
  if stopm>0 and stopd<=0 then do;
    stopd=15; flag4=1; spflag=1; end;
  if stopm<=0 and stopd>0 then do;
    stopm=6; flag5=1; spflag=1; end;
  if stopm<=0 and stopd<=0 then do;
    stopm=6; stopd=15; flag6=1; spflag=1; end;
end;
end;

```

if stopm>0 and stopd<=0 then do;
 stopd=15; flag4=1; spflag=1; end;
 if stopm<=0 and stopd>0 then do;
 stopm=6; flag5=1; spflag=1; end;
 if stopm<=0 and stopd<=0 then do;
 stopm=6; stopd=15; flag6=1; spflag=1; end;
end;
end;

/*Convert **start** month and day to total days*/
do over startm;
 if startm>0 and startd>0 then do;
 if startm=1 then sttdays=startd;
 if startm=2 then sttdays=startd+31;
 if startm=3 then sttdays=startd+59;
 if startm=4 then sttdays=startd+90;
 if startm=5 then sttdays=startd+120;
 if startm=6 then sttdays=startd+151;
 if startm=7 then sttdays=startd+181;
 if startm=8 then sttdays=startd+212;
 if startm=9 then sttdays=startd+243;
 if startm=10 then sttdays=startd+273;
 if startm=11 then sttdays=startd+304;
 if startm=12 then sttdays=startd+334;
 end;
end;

end;
end;

/**Account for leap years*/
do over starty;
 if starty=1980 or starty=1984 or starty=1988 or
 starty=1992 or starty=1996 then do;
 if startm>0 and startd>0 then do;
 if startm=1 then sttdays=startd;
 if startm=2 then sttdays=startd+31;
 if startm=3 then sttdays=startd+60;
 if startm=4 then sttdays=startd+91;
 if startm=5 then sttdays=startd+121;
 if startm=6 then sttdays=startd+152;
 if startm=7 then sttdays=startd+182;

```

if startm=8 then sttdays=startd+213;
if startm=9 then sttdays=startd+244;
if startm=10 then sttdays=startd+274;
if startm=11 then sttdays=startd+305;
if startm=12 then sttdays=startd+335;
end;
end;
end;

/**Convert stop month and day to total days*/
do over stopm;
if stopm>0 and stopd>0 then do;
if stopm=1 then stpdays=stopd;
if stopm=2 then stpdays=stopd+31;
if stopm=3 then stpdays=stopd+59;
if stopm=4 then stpdays=stopd+90;
if stopm=5 then stpdays=stopd+120;
if stopm=6 then stpdays=stopd+151;
if stopm=7 then stpdays=stopd+181;
if stopm=8 then stpdays=stopd+212;
if stopm=9 then stpdays=stopd+243;
if stopm=10 then stpdays=stopd+273;
if stopm=11 then stpdays=stopd+304;
if stopm=12 then stpdays=stopd+334;
end;
end;

/*Account for leap years*/
do over stopy;
if stopy=1980 or stopy=1984 or stopy=1988 or
    stopy=1992 or stopy=1996 then do;
if stopm>0 and stopd>0 then do;
if stopm=1 then stpdays=stopd;
if stopm=2 then stpdays=stopd+31;
if stopm=3 then stpdays=stopd+60;
if stopm=4 then stpdays=stopd+91;
if stopm=5 then stpdays=stopd+121;
if stopm=6 then stpdays=stopd+152;
if stopm=7 then stpdays=stopd+182;
if stopm=8 then stpdays=stopd+213;
if stopm=9 then stpdays=stopd+244;
if stopm=10 then stpdays=stopd+274;
if stopm=11 then stpdays=stopd+305;
if stopm=12 then stpdays=stopd+335;
end;
end;
end;

/**Convert days into week numbers*/
/**The basic formula for this is:
weekno=endweek{specific year}+
ceil[(totdays+[# of days remaining in DEC})/7] */
do over starty;
if starty>0 and sttdays>0 then do;
if starty=1980 then do;
startwk=ceil((sttdays+2)/7); end;
if starty=1981 then do;
startwk=52+ceil((sttdays+4)/7); end;
if starty=1982 then do;
startwk=104+ceil((sttdays+5)/7); end;
if starty=1983 then do;
startwk=156+ceil((sttdays+6)/7); end;
if starty=1984 then do;
startwk=209+ceil((sttdays)/7); end;
if starty=1985 then do;
startwk=261+ceil((sttdays+2)/7); end;
if starty=1986 then do;
startwk=313+ceil((sttdays+3)/7); end;
if starty=1987 then do;
startwk=365+ceil((sttdays+4)/7); end;
if starty=1988 then do;
startwk=417+ceil((sttdays+5)/7); end;
if starty=1989 then do;
startwk=470+ceil((sttdays)/7); end;
if starty=1990 then do;
startwk=522+ceil((sttdays+1)/7); end;
if starty=1991 then do;
startwk=574+ceil((sttdays+2)/7); end;
if starty=1992 then do;
startwk=626+ceil((sttdays+3)/7); end;
if starty=1993 then do;
startwk=678+ceil((sttdays+5)/7); end;
if starty=1994 then do;
startwk=730+ceil((sttdays+6)/7); end;
if starty=1995 then do;
startwk=783+ceil((sttdays)/7); end;
if starty=1996 then do;
startwk=835+ceil((sttdays+1)/7); end;
if starty=1997 then do;
startwk=887+ceil((sttdays+3)/7); end;
if starty=1998 then do;
startwk=939+ceil((sttdays+4)/7); end;
end;
if starty<0 and starty>-4 then do;
startwk=starty; end;
end;

do over stopy;
if stopy>0 and stpdays>0 then do;
if stopy=1980 then do;
stopwk=ceil((stpdays+2)/7); end;
if stopy=1981 then do;
stopwk=52+ceil((stpdays+4)/7); end;
if stopy=1982 then do;

```

```

stopwk=104+ceil((stpdays+5)/7); end;
if stopy=1983 then do;
  stopwk=156+ceil((stpdays+6)/7); end;
if stopy=1984 then do;
  stopwk=209+ceil((stpdays)/7); end;
if stopy=1985 then do;
  stopwk=261+ceil((stpdays+2)/7); end;
if stopy=1986 then do;
  stopwk=313+ceil((stpdays+3)/7); end;
if stopy=1987 then do;
  stopwk=365+ceil((stpdays+4)/7); end;
if stopy=1988 then do;
  stopwk=417+ceil((stpdays+5)/7); end;
if stopy=1989 then do;
  stopwk=470+ceil((stpdays)/7); end;
if stopy=1990 then do;
  stopwk=522+ceil((stpdays+1)/7); end;
if stopy=1991 then do;
  stopwk=574+ceil((stpdays+2)/7); end;
if stopy=1992 then do;
  stopwk=626+ceil((stpdays+3)/7); end;
if stopy=1993 then do;
  stopwk=678+ceil((stpdays+5)/7); end;
if stopy=1994 then do;
  stopwk=730+ceil((stpdays+6)/7); end;
if stopy=1995 then do;
  stopwk=783+ceil((stpdays)/7); end;
if stopy=1996 then do;
  stopwk=835+ceil((stpdays+1)/7); end;
if stopy=1997 then do;
  stopwk=887+ceil((stpdays+3)/7); end;
if stopy=1998 then do;
  stopwk=939+ceil((stpdays+4)/7); end;
end;
if stopy<0 and stopy>-4 then do;
  stopwk=stopy; end;
end;

/** This intermediate part of the program reads in raw data on within-job gaps and converts the gap dates into NLSY97 week numbers. The converted data has been suitably transformed to account for work within the week of the gap start/stop. **/


***** JOB 1 GAPS *****
/* These variables are read as follows:
BGDY1_1 = Begin day of within-job gap 1
on job 1
EGMO1_6 = End month of within-job gap 6
on job 1
BGAP1_3 = Begin week of within-job gap 3
on job 1 [CREATED] */

```

```

array bgdy (i) BGDY1_1-BGDY1_12;
array bgmo (i) BGMO1_1-BGMO1_12;
array bgyr (i) BGYR1_1-BGYR1_12;
array egdy (i) EGDY1_1-EGDY1_12;
array egmo (i) EGMO1_1-EGMO1_12;
array egyr (i) EGYR1_1-EGYR1_12;

array bdays (i) bday1_1-bday1_12;
/* begin day of job1_gap# (internal calc.)*/
array edays (i) eday1_1-eday1_12;
/* end day of job1_gap# (internal calc.)*/
array bweek (i) bgap1_1-bgap1_12;
/* begin week of job1_gap# (created) */
array eweek (i) egap1_1-egap1_12;
/* end week of job1_gap# (created) */
array bflag (i) bflg1_1-bflg1_12;
array eflag (i) eflg1_1-eflg1_12;

/* ONLY IMPUTE START/STOP DATES IF DAY IS MISSING */
***Fill-in start day for those missing;
do over bgyr;
  if bgyr>0 then do;
    if bgmo>0 and bgdy<=0 then do;
      bgdy=15; bflag=1; end;
    end;
  end;

***Fill-in stop day for those missing;
do over egyr;
  if egyr>0 then do;
    if egmo>0 and egdy<=0 then do;
      egdy=15; eflag=1; end;
    end;
  end;

***Identify within-job gaps on JOB 1 ***;
***Convert gap dates to week numbers ***;
***Convert START month and day to total days (BDAYS);

do over bgmo;
  if bgmo>0 and bgdy>0 then do;
    if bgmo=1 then bdays=bgdy;
    if bgmo=2 then bdays=bgdy+31;
    if bgmo=3 then bdays=bgdy+59;
    if bgmo=4 then bdays=bgdy+90;
    if bgmo=5 then bdays=bgdy+120;
    if bgmo=6 then bdays=bgdy+151;
    if bgmo=7 then bdays=bgdy+181;
    if bgmo=8 then bdays=bgdy+212;

```

```

if bgmo=9 then bdays=bgdy+243;
if bgmo=10 then bdays=bgdy+273;
if bgmo=11 then bdays=bgdy+304;
if bgmo=12 then bdays=bgdy+334;
end;
end;

***Account for leap years;
do over bgyr;
if bgyr=1980 or bgyr=1984 or bgyr=1988 or
   bgyr=1992 or bgyr=1996 then do;
if bgmo>0 and bgdy>0 then do;
  if bgmo=1 then bdays=bgdy;
  if bgmo=2 then bdays=bgdy+31;
  if bgmo=3 then bdays=bgdy+60;
  if bgmo=4 then bdays=bgdy+91;
  if bgmo=5 then bdays=bgdy+121;
  if bgmo=6 then bdays=bgdy+152;
  if bgmo=7 then bdays=bgdy+182;
  if bgmo=8 then bdays=bgdy+213;
  if bgmo=9 then bdays=bgdy+244;
  if bgmo=10 then bdays=bgdy+274;
  if bgmo=11 then bdays=bgdy+305;
  if bgmo=12 then bdays=bgdy+335;
end;
end;
end;

***Convert STOP month and day to total days
(EDAYS);
do over egmo;
if egmo>0 and egdy>0 then do;
  if egmo=1 then edays=egdy;
  if egmo=2 then edays=egdy+31;
  if egmo=3 then edays=egdy+59;
  if egmo=4 then edays=egdy+90;
  if egmo=5 then edays=egdy+120;
  if egmo=6 then edays=egdy+151;
  if egmo=7 then edays=egdy+181;
  if egmo=8 then edays=egdy+212;
  if egmo=9 then edays=egdy+243;
  if egmo=10 then edays=egdy+273;
  if egmo=11 then edays=egdy+304;
  if egmo=12 then edays=egdy+334;
end;
end;

***Account for leap years;
do over egyr;
if egyr=1980 or egyr=1984 or egyr=1988 or
   egyr=1992 or egyr=1996 then do;
if egmo>0 and egdy>0 then do;

```

```

if egmo=1 then edays=egdy;
if egmo=2 then edays=egdy+31;
if egmo=3 then edays=egdy+60;
if egmo=4 then edays=egdy+91;
if egmo=5 then edays=egdy+121;
if egmo=6 then edays=egdy+152;
if egmo=7 then edays=egdy+182;
if egmo=8 then edays=egdy+213;
if egmo=9 then edays=egdy+244;
if egmo=10 then edays=egdy+274;
if egmo=11 then edays=egdy+305;
if egmo=12 then edays=egdy+335;
end;
end;
end;

/** Convert days into week numbers **/
/** The basic formula for this is:
weekno=endweek{specific year}+
ceil[(totdays+[# of days remaining in DEC])/7]*/

```

/* Note: This program takes the week following the actual start of the gap as the measure of when the non-working period begins. */

```

do over bgyr;
if bgyr>0 and bdays>0 then do;
if bgyr=1980 then do;
  bweek=ceil((bdays+2)/7);
  bweek=bweek+1; end;
if bgyr=1981 then do;
  bweek=52+ceil((bdays+4)/7);
  bweek=bweek+1; end;
if bgyr=1982 then do;
  bweek=104+ceil((bdays+5)/7);
  bweek=bweek+1; end;
if bgyr=1983 then do;
  bweek=156+ceil((bdays+6)/7);
  bweek=bweek+1; end;
if bgyr=1984 then do;
  bweek=209+ceil((bdays)/7);
  bweek=bweek+1; end;
if bgyr=1985 then do;
  bweek=261+ceil((bdays+2)/7);
  bweek=bweek+1; end;
if bgyr=1986 then do;
  bweek=313+ceil((bdays+3)/7);
  bweek=bweek+1; end;
if bgyr=1987 then do;
  bweek=365+ceil((bdays+4)/7);
  bweek=bweek+1; end;
if bgyr=1988 then do;
  bweek=417+ceil((bdays+5)/7);

```

<pre> bweek=bweek+1; end; if bgyr=1989 then do; bweek=470+ceil((bdays)/7); bweek=bweek+1; end; if bgyr=1990 then do; bweek=522+ceil((bdays+1)/7); bweek=bweek+1; end; if bgyr=1991 then do; bweek=574+ceil((bdays+2)/7); bweek=bweek+1; end; if bgyr=1992 then do; bweek=626+ceil((bdays+3)/7); bweek=bweek+1; end; if bgyr=1993 then do; bweek=678+ceil((bdays+5)/7); bweek=bweek+1; end; if bgyr=1994 then do; bweek=730+ceil((bdays+6)/7); bweek=bweek+1; end; if bgyr=1995 then do; bweek=783+ceil((bdays)/7); bweek=bweek+1; end; if bgyr=1996 then do; bweek=835+ceil((bdays+1)/7); bweek=bweek+1; end; if bgyr=1997 then do; bweek=887+ceil((bdays+3)/7); bweek=bweek+1; end; if bgyr=1998 then do; bweek=939+ceil((bdays+4)/7); bweek=bweek+1; end; end; end; do over egyr; if egyr>0 and edays>0 then do; if egyr=1980 then do; eweek=ceil((edays+2)/7); eweek=eweek-1; end; if egyr=1981 then do; eweek=52+ceil((edays+4)/7); eweek=eweek-1; end; if egyr=1982 then do; eweek=104+ceil((edays+5)/7); eweek=eweek-1; end; if egyr=1983 then do; eweek=156+ceil((edays+6)/7); eweek=eweek-1; end; if egyr=1984 then do; eweek=209+ceil((edays)/7); eweek=eweek-1; end; if egyr=1985 then do; </pre>	<pre> eweek=261+ceil((edays+2)/7); eweek=eweek-1; end; if egyr=1986 then do; eweek=313+ceil((edays+3)/7); eweek=eweek-1; end; if egyr=1987 then do; eweek=365+ceil((edays+4)/7); eweek=eweek-1; end; if egyr=1988 then do; eweek=417+ceil((edays+5)/7); eweek=eweek-1; end; if egyr=1989 then do; eweek=470+ceil((edays)/7); eweek=eweek-1; end; if egyr=1990 then do; eweek=522+ceil((edays+1)/7); eweek=eweek-1; end; if egyr=1991 then do; eweek=574+ceil((edays+2)/7); eweek=eweek-1; end; if egyr=1992 then do; eweek=626+ceil((edays+3)/7); eweek=eweek-1; end; if egyr=1993 then do; eweek=678+ceil((edays+5)/7); eweek=eweek-1; end; if egyr=1994 then do; eweek=730+ceil((edays+6)/7); eweek=eweek-1; end; if egyr=1995 then do; eweek=783+ceil((edays)/7); eweek=eweek-1; end; if egyr=1996 then do; eweek=835+ceil((edays+1)/7); eweek=eweek-1; end; if egyr=1997 then do; eweek=887+ceil((edays+3)/7); eweek=eweek-1; end; if egyr=1998 then do; eweek=939+ceil((edays+4)/7); eweek=eweek-1; end; end; end; ***At this point the program loops through the same code for gaps in jobs 2-6 (there are no gaps for job 7). This code is eliminated here due to space restrictions. Users who need access to the entire program should contact NLS User Services.***/ </pre>
--	--

```
*****The program next calculates the total
number of weeks worked on each job.*****/

array job1wks (i) wk1_1-wk1_992;
array job2wks (i) wk2_1-wk2_992;
array job3wks (i) wk3_1-wk3_992;
array job4wks (i) wk4_1-wk4_992;
array job5wks (i) wk5_1-wk5_992;
array job6wks (i) wk6_1-wk6_992;
array job7wks (i) wk7_1-wk7_992;
array alljobs (i) wks1-wks992;
array starw (i) starw1-starw7;
array stopw (i) stopw1-stopw7;

/* Default Settings */

do i=1 to 992;
    job1wks=0; job2wks=0; job3wks=0;
    job4wks=0; job5wks=0; job6wks=0;
    job7wks=0;
end;

***** Total Weeks Worked on Job 1 *****

if starw1>0 & stopw1>0 then do; /* [1] */
    do i=(starw1) to (stopw1);
        job1wks=1;
    end;

    ** Remove gap 1 on job 1;
    if bgap1_1>0 & egap1_1>0 then do;
        do i=(bgap1_1) to (egap1_1);
            job1wks=0; end;
    end;

    ** Remove gap 1 on job 1 - begin gap date bad;
    if bgfl1_1=1 & egap1_1>0 then do;
        do i=(starw1) to (egap1_1);
            job1wks=-3; gpfl1_1=1; end;
    end;

    ** Remove gap 1 on job 1 - end gap date bad;
    if bgap1_1>0 & egfl1_1=1 then do;
        do i=(bgap1_1) to (stopw1);
            job1wks=-3; gpfl1_1=1; end;
    end;

    ** Remove gap 1 on job 1 - both gap dates bad;
    if bgfl1_1=1 & egfl1_1=1 then do;
        do i=(starw1) to (stopw1);
            job1wks=-3; gpfl1_1=1; end;
    end;

    ** Remove gap 2 on job 1;

```

```
if bgap1_2>0 & egap1_2>0 then do;
    do i=(bgap1_2) to (egap1_2);
        job1wks=0; end;
end;

** Remove gap 2 on job 1 - begin gap date bad;
if bgfl1_2=1 & egap1_2>0 then do;
    do i=(starw1) to (egap1_2);
        job1wks=-3; gpfl1_2=1; end;
end;

** Remove gap 2 on job 1 - end gap date bad;
if bgap1_2>0 & egfl1_2=1 then do;
    do i=(bgap1_2) to (stopw1);
        job1wks=-3; gpfl1_2=1; end;
end;

** Remove gap 2 on job 1 - both gap dates bad;
if bgfl1_2=1 & egfl1_2=1 then do;
    do i=(starw1) to (stopw1);
        job1wks=-3; gpfl1_2=1; end;
end;

** Remove gap 3 on job 1;
if bgap1_3>0 & egap1_3>0 then do;
    do i=(bgap1_3) to (egap1_3);
        job1wks=0; end;
end;

** Remove gap 3 on job 1 - begin gap date bad;
if bgfl1_3=1 & egap1_3>0 then do;
    do i=(starw1) to (egap1_3);
        job1wks=-3; gpfl1_3=1; end;
end;

** Remove gap 3 on job 1 - end gap date bad;
if bgap1_3>0 & egfl1_3=1 then do;
    do i=(bgap1_3) to (stopw1);
        job1wks=-3; gpfl1_3=1; end;
end;

** Remove gap 3 on job 1 - both gap dates bad;
if bgfl1_3=1 & egfl1_3=1 then do;
    do i=(starw1) to (stopw1);
        job1wks=-3; gpfl1_3=1; end;
end;

** Remove gap 4 on job 1;
if bgap1_4>0 & egap1_4>0 then do;
    do i=(bgap1_4) to (egap1_4);
        job1wks=0; end;
end;

** Remove gap 4 on job 1 - begin gap date bad;
if bgfl1_4=1 & egap1_4>0 then do;
    do i=(starw1) to (egap1_4);

```

```

        job1wks=-3; gpfl1_4=1; end;
      end;
** Remove gap 4 on job 1 - end gap date bad;
if bgap1_4>0 & egfl1_4=1 then do;
  do i=(bgap1_4) to (stopw1);
    job1wks=-3; gpfl1_4=1; end;
  end;
** Remove gap 4 on job 1 - both gap dates bad;
if bgfl1_4=1 & egfl1_4=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_4=1; end;
  end;

** Remove gap 5 on job 1;
if bgap1_5>0 & egap1_5>0 then do;
  do i=(bgap1_5) to (egap1_5);
    job1wks=0; end;
  end;

** Remove gap 5 on job 1 – begin gap date bad;
if bgfl1_5=1 & egap1_5>0 then do;
  do i=(starw1) to (egap1_5);
    job1wks=-3; gpfl1_5=1; end;
  end;
** Remove gap 5 on job 1 - end gap date bad;
if bgap1_5>0 & egfl1_5=1 then do;
  do i=(bgap1_5) to (stopw1);
    job1wks=-3; gpfl1_5=1; end;
  end;
** Remove gap 5 on job 1 - both gap dates bad;
if bgfl1_5=1 & egfl1_5=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_5=1; end;
  end;

** Remove gap 6 on job 1;
if bgap1_6>0 & egap1_6>0 then do;
  do i=(bgap1_6) to (egap1_6);
    job1wks=0; end;
  end;

** Remove gap 6 on job 1 - begin gap date bad;
if bgfl1_6=1 & egap1_6>0 then do;
  do i=(starw1) to (egap1_6);
    job1wks=-3; gpfl1_6=1; end;
  end;
** Remove gap 6 on job 1 - end gap date bad;
if bgap1_6>0 & egfl1_6=1 then do;
  do i=(bgap1_6) to (stopw1);
    job1wks=-3; gpfl1_6=1; end;
  end;
** Remove gap 6 on job 1 - both gap dates bad;
      if bgfl1_6=1 & egfl1_6=1 then do;
        do i=(starw1) to (stopw1);
          job1wks=-3; gpfl1_6=1; end;
        end;

        if bgfl1_6=1 & egfl1_6=1 then do;
          do i=(starw1) to (stopw1);
            job1wks=-3; gpfl1_6=1; end;
          end;

** Remove gap 7 on job 1;
if bgap1_7>0 & egap1_7>0 then do;
  do i=(bgap1_7) to (egap1_7);
    job1wks=0; end;
  end;

** Remove gap 7 on job 1 - begin gap date bad;
if bgfl1_7=1 & egap1_7>0 then do;
  do i=(starw1) to (egap1_7);
    job1wks=-3; gpfl1_7=1; end;
  end;
** Remove gap 7 on job 1 - end gap date bad;
if bgap1_7>0 & egfl1_7=1 then do;
  do i=(bgap1_7) to (stopw1);
    job1wks=-3; gpfl1_7=1; end;
  end;
** Remove gap 7 on job 1 - both gap dates bad;
if bgfl1_7=1 & egfl1_7=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_7=1; end;
  end;

** Remove gap 8 on job 1;
if bgap1_8>0 & egap1_8>0 then do;
  do i=(bgap1_8) to (egap1_8);
    job1wks=0; end;
  end;

** Remove gap 8 on job 1 - begin gap date bad;
if bgfl1_8=1 & egap1_8>0 then do;
  do i=(starw1) to (egap1_8);
    job1wks=-3; gpfl1_8=1; end;
  end;
** Remove gap 8 on job 1 - end gap date bad;
if bgap1_8>0 & egfl1_8=1 then do;
  do i=(bgap1_8) to (stopw1);
    job1wks=-3; gpfl1_8=1; end;
  end;
** Remove gap 8 on job 1 - both gap dates bad;
if bgfl1_8=1 & egfl1_8=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_8=1; end;
  end;

** Remove gap 9 on job 1;
if bgap1_9>0 & egap1_9>0 then do;
  do i=(bgap1_9) to (egap1_9);
    job1wks=0; end;
  end;

```

```

job1wks=0; end;
end;

** Remove gap 9 on job 1 - begin gap date bad;
if bgfl1_9=1 & egap1_9>0 then do;
  do i=(starw1) to (egap1_9);
    job1wks=-3; gpfl1_9=1; end;
  end;
** Remove gap 9 on job 1 - end gap date bad;
if bgap1_9>0 & egfl1_9=1 then do;
  do i=(bgap1_9) to (stopw1);
    job1wks=-3; gpfl1_9=1; end;
  end;
** Remove gap 9 on job 1 - both gap dates bad;
if bgfl1_9=1 & egfl1_9=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_9=1; end;
  end;

** Remove gap 10 on job 1;
if bgap1_10>0 & egap1_10>0 then do;
  do i=(bgap1_10) to (egap1_10);
    job1wks=0; end;
  end;

** Remove gap 10 on job 1 - begin gap date bad;
if bgfl1_10=1 & egap1_10>0 then do;
  do i=(starw1) to (egap1_10);
    job1wks=-3; gpfl1_10=1; end;
  end;
** Remove gap 10 on job 1 - end gap date bad;
if bgap1_10>0 & egfl1_10=1 then do;
  do i=(bgap1_10) to (stopw1);
    job1wks=-3; gpfl1_10=1; end;
  end;
** Remove gap 10 on job 1 - both gap dates bad;
if bgfl1_10=1 & egfl1_10=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_10=1; end;
  end;

do i=(starw1) to (stopw1);
  job1wks=-3; gpfl1_10=1; end;
end;

** Remove gap 11 on job 1;
if bgap1_11>0 & egap1_11>0 then do;
  do i=(bgap1_11) to (egap1_11);
    job1wks=0; end;
  end;

** Remove gap 11 on job 1 – begin gap date bad;
if bgfl1_11=1 & egap1_11>0 then do;
  do i=(starw1) to (egap1_11);
    job1wks=-3; gpfl1_11=1; end;
  end;
** Remove gap 11 on job 1 - end gap date bad;
if bgap1_11>0 & egfl1_11=1 then do;
  do i=(bgap1_11) to (stopw1);
    job1wks=-3; gpfl1_11=1; end;
  end;
** Remove gap 11 on job 1 - both gap dates bad;
if bgfl1_11=1 & egfl1_11=1 then do;
  do i=(starw1) to (stopw1);
    job1wks=-3; gpfl1_11=1; end;
  end;

end; /* [1] */

***** At this point the program repeats the
same code to calculate total weeks worked for jobs
2–7. The only difference is that the number of
gaps is smaller for some of these jobs. Users who
need to see the complete code should contact NLS
User Services. *****

endsas;

```

BDATE1.SAS

```

/* This program creates birthwk (default=0 birth date < 12/30/79) and age14wk (default=9999 if age 14
date > 1/1/99). */
/** Calculate Age 14 year **/
AGE14YR=birthyr+14;

/** Convert Age 14 Birthdate to week # **/
/**Convert age14 month and day to total days
(bbdays)**/
if birthmo>0 and birthdy>0 then do;
  if birthmo=1 then bbdays=birthdy;
  if birthmo=2 then bbdays=birthdy+31;
  if birthmo=3 then bbdays=birthdy+59;
  if birthmo=4 then bbdays=birthdy+90;
  if birthmo=5 then bbdays=birthdy+120;
  if birthmo=6 then bbdays=birthdy+151;
  if birthmo=7 then bbdays=birthdy+181;
  if birthmo=8 then bbdays=birthdy+212;
  if birthmo=9 then bbdays=birthdy+243;
  if birthmo=10 then bbdays=birthdy+273;
  if birthmo=11 then bbdays=birthdy+304;

```

```

if birthmo=12 then bbdays=birthdy+334;
end;

***Account for leap years;
if age14yr=1980 or age14yr=1984 or
    age14yr=1988 or age14yr=1992 or
    age14yr=1996 then do;
if birthmo>0 and birthdy>0 then do;
    if birthmo=1 then bbdays=birthdy;
    if birthmo=2 then bbdays=birthdy+31;
    if birthmo=3 then bbdays=birthdy+60;
    if birthmo=4 then bbdays=birthdy+91;
    if birthmo=5 then bbdays=birthdy+121;
    if birthmo=6 then bbdays=birthdy+152;
    if birthmo=7 then bbdays=birthdy+182;
    if birthmo=8 then bbdays=birthdy+213;
    if birthmo=9 then bbdays=birthdy+244;
    if birthmo=10 then bbdays=birthdy+274;
    if birthmo=11 then bbdays=birthdy+305;
    if birthmo=12 then bbdays=birthdy+335;
end;
end;

/**Convert days into week numbers*/
/** The basic formula for this is:
weekno=endweek{specific year}+
ceil[(totdays+[# of days remaining in DEC])/7] */

/* Default age 14 week = 9999 */
age14wk=9999;

if age14yr>0 and bbdays>0 then do;
if age14yr=1980 then do;
    age14wk=ceil((bbdays+2)/7); end;
if age14yr=1981 then do;
    age14wk=52+ceil((bbdays+4)/7); end;
if age14yr=1982 then do;
    age14wk=104+ceil((bbdays+5)/7); end;
if age14yr=1983 then do;
    age14wk=156+ceil((bbdays+6)/7); end;
if age14yr=1984 then do;
    age14wk=209+ceil((bbdays)/7); end;
if age14yr=1985 then do;
    age14wk=261+ceil((bbdays+2)/7); end;
if age14yr=1986 then do;
    age14wk=313+ceil((bbdays+3)/7); end;
if age14yr=1987 then do;
    age14wk=365+ceil((bbdays+4)/7); end;
if age14yr=1988 then do;
    age14wk=417+ceil((bbdays+5)/7); end;
if age14yr=1989 then do;
    age14wk=470+ceil((bbdays)/7); end;

if age14yr=1990 then do;
    age14wk=522+ceil((bbdays+1)/7); end;
if age14yr=1991 then do;
    age14wk=574+ceil((bbdays+2)/7); end;
if age14yr=1992 then do;
    age14wk=626+ceil((bbdays+3)/7); end;
if age14yr=1993 then do;
    age14wk=678+ceil((bbdays+5)/7); end;
if age14yr=1994 then do;
    age14wk=730+ceil((bbdays+6)/7); end;
if age14yr=1995 then do;
    age14wk=783+ceil((bbdays)/7); end;
if age14yr=1996 then do;
    age14wk=835+ceil((bbdays+1)/7); end;
if age14yr=1997 then do;
    age14wk=887+ceil((bbdays+3)/7); end;
if age14yr=1998 then do;
    age14wk=939+ceil((bbdays+4)/7); end;
end;

/** Calulate Age 16 year **/
AGE16YR=birthyr+16;

/** Convert Age 16 birthdate to week # */
/**Convert age16 month and day to total days
(bbdays)**/
if birthmo>0 and birthdy>0 then do;
    if birthmo=1 then bbdays=birthdy;
    if birthmo=2 then bbdays=birthdy+31;
    if birthmo=3 then bbdays=birthdy+59;
    if birthmo=4 then bbdays=birthdy+90;
    if birthmo=5 then bbdays=birthdy+120;
    if birthmo=6 then bbdays=birthdy+151;
    if birthmo=7 then bbdays=birthdy+181;
    if birthmo=8 then bbdays=birthdy+212;
    if birthmo=9 then bbdays=birthdy+243;
    if birthmo=10 then bbdays=birthdy+273;
    if birthmo=11 then bbdays=birthdy+304;
    if birthmo=12 then bbdays=birthdy+334;
end;

/**Account for leap years*/
if age16yr=1980 or age16yr=1984 or
    age16yr=1988 or age16yr=1992 or
    age16yr=1996 then do;
if birthmo>0 and birthdy>0 then do;
    if birthmo=1 then bbdays=birthdy;
    if birthmo=2 then bbdays=birthdy+31;
    if birthmo=3 then bbdays=birthdy+60;
    if birthmo=4 then bbdays=birthdy+91;
    if birthmo=5 then bbdays=birthdy+121;
    if birthmo=6 then bbdays=birthdy+152;
```

```

if birthmo=7 then bbdays=birthdy+182;
if birthmo=8 then bbdays=birthdy+213;
if birthmo=9 then bbdays=birthdy+244;
if birthmo=10 then bbdays=birthdy+274;
if birthmo=11 then bbdays=birthdy+305;
if birthmo=12 then bbdays=birthdy+335;
end;
end;

/***Convert days into week numbers***/
/*** The basic formula for this is:
weekno=endweek{specific year}+
ceil[(totdays+[# of days remaining in DEC})/7] */

/* Default age 16 week = 9999 */
age16wk=9999;

if age16yr>0 and bbdays>0 then do;
if age16yr=1980 then do;
age16wk=ceil((bbdays+2)/7); end;
if age16yr=1981 then do;
age16wk=52+ceil((bbdays+4)/7); end;
if age16yr=1982 then do;
age16wk=104+ceil((bbdays+5)/7); end;
if age16yr=1983 then do;
age16wk=156+ceil((bbdays+6)/7); end;
if age16yr=1984 then do;
age16wk=209+ceil((bbdays)/7); end;
if age16yr=1985 then do;
age16wk=261+ceil((bbdays+2)/7); end;
if age16yr=1986 then do;
age16wk=313+ceil((bbdays+3)/7); end;
if age16yr=1987 then do;
age16wk=365+ceil((bbdays+4)/7); end;
if age16yr=1988 then do;
age16wk=417+ceil((bbdays+5)/7); end;
if age16yr=1989 then do;
age16wk=470+ceil((bbdays)/7); end;
if age16yr=1990 then do;
age16wk=522+ceil((bbdays+1)/7); end;
if age16yr=1991 then do;
age16wk=574+ceil((bbdays+2)/7); end;
if age16yr=1992 then do;
age16wk=626+ceil((bbdays+3)/7); end;
if age16yr=1993 then do;
age16wk=678+ceil((bbdays+5)/7); end;
if age16yr=1994 then do;
age16wk=730+ceil((bbdays+6)/7); end;
if age16yr=1995 then do;
age16wk=783+ceil((bbdays)/7); end;
if age16yr=1996 then do;
age16wk=835+ceil((bbdays+1)/7); end;

```

```

if age16yr=1997 then do;
age16wk=887+ceil((bbdays+3)/7); end;
if age16yr=1998 then do;
age16wk=939+ceil((bbdays+4)/7); end;
end;

/*** Convert Birthdate to week number ***/
/* Default birthdate week=0 if birthdate <
12/30/79 */
birthwk=0;

if birthyr>0 and bbdays>0 then do;
if birthyr=1980 then do;
birthwk=ceil((bbdays+2)/7); end;
if birthyr=1981 then do;
birthwk=52+ceil((bbdays+4)/7); end;
if birthyr=1982 then do;
birthwk=104+ceil((bbdays+5)/7); end;
if birthyr=1983 then do;
birthwk=156+ceil((bbdays+6)/7); end;
if birthyr=1984 then do;
birthwk=209+ceil((bbdays)/7); end;
if birthyr=1985 then do;
birthwk=261+ceil((bbdays+2)/7); end;
if birthyr=1986 then do;
birthwk=313+ceil((bbdays+3)/7); end;
if birthyr=1987 then do;
birthwk=365+ceil((bbdays+4)/7); end;
if birthyr=1988 then do;
birthwk=417+ceil((bbdays+5)/7); end;
if birthyr=1989 then do;
birthwk=470+ceil((bbdays)/7); end;
if birthyr=1990 then do;
birthwk=522+ceil((bbdays+1)/7); end;
if birthyr=1991 then do;
birthwk=574+ceil((bbdays+2)/7); end;
if birthyr=1992 then do;
birthwk=626+ceil((bbdays+3)/7); end;
if birthyr=1993 then do;
birthwk=678+ceil((bbdays+5)/7); end;
if birthyr=1994 then do;
birthwk=730+ceil((bbdays+6)/7); end;
if birthyr=1995 then do;
birthwk=783+ceil((bbdays)/7); end;
if birthyr=1996 then do;
birthwk=835+ceil((bbdays+1)/7); end;
if birthyr=1997 then do;
birthwk=887+ceil((bbdays+3)/7); end;
if birthyr=1998 then do;
birthwk=835+ceil((bbdays+1)/7); end;
endsas;

```

YOUTH'S EMPLOYMENT STATUS RECODE

Variables Created: CV_ESR
CV_ESR_COLLAPSED

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD	Name in Program	Question Name on CD
C2100	YCPS-2100	C87001	YCPS-87001	C15800	YCPS-15800
C2300	YCPS-2300	C87002	YCPS-87002	C15900	YCPS-15900
C2400	YCPS-2400	C96001	YCPS-96001	C176001	YCPS-176001
C2600	YCPS-2600	C96002	YCPS-96002	C176002	YCPS-176002
C2800	YCPS-2800	C10400	YCPS-10400	C186001	YCPS-186001
C2900	YCPS-2900	C106001	YCPS-106001	C186002	YCPS-186002
C3100	YCPS-3100	C106002	YCPS-106002	C23600	YCPS-23600
C78001	YCPS-78001	C11000	YCPS-11000	C23700	YCPS-23700
C78002	YCPS-78002	C14000	YCPS-14000	C27300	YCPS-27300
C79001	YCPS-79001	C14100	YCPS-14100	C30700	YCPS-30700
C79002	YCPS-79002	C14200	YCPS-14200	C30800	YCPS-30800
C82001	YCPS-82001	C15200	YCPS-15200	PUBID	PUBID
C82002	YCPS-82002	C15300	YCPS-15300		

Codes for Created Variable

Employment Status Recode

- 1 = at work
- 2 = with job, not at work
- 3 = layoff, unemployed
- 4 = look for work, unemployed
- 5= not in labor force, retired
- 6 = not in labor force, disabled (long-term physical or mental illness, lasting six months or longer)
- 7 = not in labor force, other
- 8 = in active Armed Forces

Collapsed Employment Status Recode

- 1 = employed
- 2 = unemployed
- 3 = not in the labor force
- 4 = in active Armed Forces

This program creates two employment status variables: employment status recode (ESR) and collapsed ESR. These variables are created only for respondents age 15 or older as of the interview date. Note that enlistment in the active Armed Forces takes precedence over concurrent civilian employment.

```
/* First create the ESR variable: */
/*Initialize esr*/
esr=-1;
if C2100=-2 then esr=-2;
if C2100=-3 then esr=-3;
if C2100=0 or C2100=-4 then esr=-4;

/*Define some arrays to be used later in the program at various times */
array C7800 C78001 C78002;
array C7900 C79001 C79002;
array C8200 C82001 C82002;
array C8700 C87001 C87002;
array C9600 C96001 C96002;
array C10600 C106001 C106002;
array C17600 C176001 C176002;
array C18600 C186001 C186002;
```

Appendix 2: Employment Variable Creation

```
/*Create hrck6 and hrck7 for when the hh owns a farm and when the hh does not own a farm*/  
  
hrck6=3;  
if (C15200=0 and C15900=0) and (C2800=0 or C2800=-2 or C2800=-1 or C2800=-3) then hrck6=1;  
if (C15200=0 and C15900=0) then hrck6=2;  
  
if (C10400 ge 0 and C11000 ge 0) then hrslt=C10400+C11000;  
if (C10400 ge 0 and C11000 lt 0) then hrslt=C10400;  
if (C11000 ge 0 and C10400 lt 0) then hrslt=C11000;  
  
if (C15200 ge 0 and C15900 ge 0) then hractt=C15200+C15900;  
if (C15200 ge 0 and C15900 lt 0) then hractt=C15200;  
if (C15900 ge 0 and C15200 lt 0) then hractt=C15900;  
  
hrck7=5;  
if (C2900=0 or C2900=-2 or C2900=-1 or C2900=-3) and ((C15200 lt 15 and C15200 ge 0) or C15200 eq -2) then  
    hrck7=1;  
if (C2900=0 or C2900=-2 or C2900=-1 or C2900=-3) and (C15200 ge 15) then hrck7=2;  
if (hrslt ge 35) and (hractt lt 35) and (C15200 ge 0) then hrck7=3;  
if (hrslt ge 35) and (hractt lt 35) and (C15900 ge 0) then hrck7=3;  
if (C14000=1 and hractt lt 35) and (C14100=1 or C14100=2 or C14100=3) then hrck7=4;  
  
/*If hh owns a farm and worked last week or was absent from work:*/  
if C2100=1 and C2600=1 then do;  
/* If at work:*/  
    do I=1 to 2;  
        if (hrck7 eq 2 or hrck7 eq 3 or hrck7 eq 4 or hrck7 eq 5) then do;  
            if C2900=1 or (C2900=0 and C3100=1 and C10600(I)>=15) then esr=1;  
            end;  
        end;  
/* If absent:*/  
    do I=1 to 2;  
        if (C2900=0 and C3100=0 and C8200(I)=1 and C9600(I)=1) or (C2900=0 and  
            C3100=0 and C8200(I)=1 and C9600(I)=0 and C10600(I)>=15) then esr=2;  
        end;  
    end;  
  
/*If hh does not own a farm and worked or was absent from work during last week*/  
/* If at work:*/  
if C2800=1 and hrck6 eq 3 and C2100=1 and C2600 le 1 then esr=1;  
/* If absent:*/  
do I=1 to 2;  
    if C2800=0 and C8200(I)=1 and C2100=1 and C2600 le 1 then esr=2;  
end;  
  
/*If laid off during last week*/  
do I=1 to 2;  
    if (C2100=1 and C8700(I)=1 and C17600(I)=1) then esr=3;  
end;  
  
/*If unemployed and looking for a job during last week*/  
do I=1 to 2;  
    if (C2100=1 and C18600(I)=1 and C23600=1)  
    or (C2100=1 and C18600(I)=1 and C23600=0 and C23700=1)  
    then esr=4;  
end;
```

```

/*If not in the labor force and retired */
if C2100=1 and (C2800=2 or C2900=3 or C30800=5) then esr=5;

/*If not in the labor force and disabled */
/* If hh owns a farm:*/
    do I=1 to 2;
        if C2100=1 and C2600=1 then do;
            if C2900=3 and (C7800(I)=1 or C7900(I)=1) then esr=6;
            end;
        end;
    /* If hh does not own a farm:*/
    do I=1 to 2;
        if C2100=1 and C2600=0 then do;
            if C2800=3 and (C7800(I)=1 or C7900(I)=1) then esr=6;
            end;
    end;

if esr=-1 then do;

/*If not in the labor force and "other" category:*/
if C2100=1 and (C30700=3 or C30800=1 or C30800=2 or C30800=3 or C30800=4 or C30800=6) then
esr=7;

/*If in the army during last week*/
if C2300=1 and C2400=1 then esr=8;

end;

/* Create the Collapsed ESR code */
if esr=1 or esr=2 then collesr=1;
if esr=3 or esr=4 then collesr=2;
if esr=5 or esr=6 or esr=7 then collesr=3;
if esr=8 then collesr=4;
if esr=-4 then collesr=-4;
if esr=-1 then collesr=-1;
if esr=-3 then collesr=-3;
if esr=-2 then collesr=-2;

endsas;

```

HOURLY RATE OF PAY

Variables Created: CV_HRLY_PAY

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
E1710001-07	YEMP-17100.01-.07	E5510001-05	YEMP-55100.01-.05
E1900001-07	YEMP-19000.01-.07	E5520001-04	YEMP-55200.01-.04
E1920001-07	YEMP-19200.01-.07	E5980001-07	YEMP-59800.01-.07
E2290001-06	YEMP-22900.01-.06	E6200001-07	YEMP-62000.01-.07
E2300001-07	YEMP-23000.01-.07	E6570001-04	YEMP-65700.01-.04
E2320001, 02	YEMP-23200.01, .02	E6580001-07	YEMP-65800.01-.07
E2380001, 02	YEMP-23800.01, .02	E6600001-05	YEMP-66000.01-.05
E2390001, 02	YEMP-23900.01, .02	E6670001	YEMP-66700.01
E2690002	YEMP-26900.02	E7200001-07	YEMP-72000.01-.07
E2920001-07	YEMP-29200.01-.07	E7620001-05	YEMP-76200.01-.05
E3340001-05	YEMP-33400.01-.05	E7630001, 02	YEMP-76300.01, .02
E3350001-04	YEMP-33500.01-.04	E7640001, 02	YEMP-76400.01, .02
E3360001-05	YEMP-33600.01-.05	E7800001	YEMP-18000.01
E3530001-03	YEMP-35300.01-.03	E7810001, 02	YEMP-78100.01, .02
E3560001-05	YEMP-35600.01-.05	E7840001-04	YEMP-78400.01-.04
E3610001-04	YEMP-36100.01-.04	E7900001-04	YEMP-79000.01-.04
E3620001-05	YEMP-36200.01-.05	E7920001-07	YEMP-79200.01-.07
E3780001-07	YEMP-37800.01-.07	E8220001-07	YEMP-82200.01-.07
E3790001-06	YEMP-37900.01-.06	E8310001-05	YEMP-83100.01-.05
E3820001-06	YEMP-38200.01-.06	E8680001-03	YEMP-86800.01-.03
E4190001-06	YEMP-41900.01-.06	E8690001-05	YEMP-86900.01-.05
E4200001-06	YEMP-42000.01-.06	E8710001	YEMP-87100.01
E4220003	YEMP-42200.03	E9310001-05	YEMP-93100.01-.05
E4290001	YEMP-42900.01	E9730001-05	YEMP-97300.01-.05
E4820001-06	YEMP-48200.01-.06	E9740001-04	YEMP-97400.01-.04
E5240001-05	YEMP-52400.01-.05	E9750001, 02	YEMP-97500.01, .02
E5250001-05	YEMP-52500.01-.05	E9830001, 02	YEMP-98300.01, .02
E5260001-05	YEMP-52600.01-.05	E9920001, 02	YEMP-99200.01, .02
E5340001, 02	YEMP-53400.01, .02	E9950001-04	YEMP-99500.01-.04
E5420001, 02	YEMP-54200.01, .02	E100e01-04	YEMP-100000.01-.04
E5430001, 02	YEMP-54300.01, .02	E1001e01-04	YEMP-100100.01-.04
E5460001, 05	YEMP-54600.01, .05	PUBID	PUBID

Codes for Created Variable

Note that hourly rate of pay is reported with two implied decimal places.

This program created the hourly rate of pay for NLSY97 respondents. The hourly rate of pay is constructed from stop date information for respondents who have a job lasting more than 13 weeks. For all other respondents the start wage is used.

The hourly rate of pay variable uses the following 4 reporting categories:

1. The start wage for youths under the age of 16 as of the survey date who only report a start wage;
2. The stop wage for youths under the age of 16 as of the survey date who report both a start wage and a stop wage;
3. The start wage for youths age 16 or older as of the survey date who only report a start wage; and
4. The stop wage for youths age 16 or older as of the survey date who report both a start wage and a stop wage.

/*Set up the hourly wage for youths who report their wage hourly. Both the hourly wage reported in the non-overtime and the overtime loops can be used without any further calculations.*/

ARRAY E17100 E1710001 E1710002 E1710003 E1710004 E1710005 E1710006 E1710007 ;

ARRAY E37800 E3780001 E3780002 E3780003 E3780004 E3780005 E3780006 E3780007 ;
ARRAY E79200 E7920001 E7920002 E7920003 E7920004 E7920005 E7920006 E7920007 ;
ARRAY E19000 E1900001 E1900002 E1900003 E1900004 E1900005 E1900006 E1900007 ;
ARRAY E19200 E1920001 E1920002 E1920003 E1920004 E1920005 E1920006 E1920007 ;
ARRAY E22900 E2290001 E2290002 E2290003 E2290004 E2290005 E2290006 E2290007 ;
ARRAY E23000 E2300001 E2300002 E2300003 E2300004 E2300005 E2300006 E2300007 ;
ARRAY E23200 E2320001 E2320002 E2320003 E2320004 E2320005 E2320006 E2320007 ;
ARRAY E33400 E3340001 E3340002 E3340003 E3340004 E3340005 E3340006 E3340007 ;
ARRAY E33500 E3350001 E3350002 E3350003 E3350004 E3350005 E3350006 E3350007 ;
ARRAY E33600 E3360001 E3360002 E3360003 E3360004 E3360005 E3360006 E3360007 ;
ARRAY E23900 E2390001 E2390002 E2390003 E2390004 E2390005 E2390006 E2390007 ;
ARRAY E34400 E3440001 E3440002 E3440003 E3440004 E3440005 E3440006 E3440007 ;
ARRAY E35300 E3530001 E3530002 E3530003 E3530004 E3530005 E3530006 E3530007 ;
ARRAY E35600 E3560001 E3560002 E3560003 E3560004 E3560005 E3560006 E3560007 ;
ARRAY E36100 E3610001 E3610002 E3610003 E3610004 E3610005 E3610006 E3610007 ;
ARRAY E36200 E3620001 E3620002 E3620003 E3620004 E3620005 E3620006 E3620007 ;
ARRAY E37900 E3790001 E3790002 E3790003 E3790004 E3790005 E3790006 E3790007 ;
ARRAY E38200 E3820001 E3820002 E3820003 E3820004 E3820005 E3820006 E3820007 ;
ARRAY E41900 E4190001 E4190002 E4190003 E4190004 E4190005 E4190006 E4190007 ;
ARRAY E42000 E4200001 E4200002 E4200003 E4200004 E4200005 E4200006 E4200007 ;
ARRAY E42200 E4220001 E4220002 E4220003 E4220004 E4220005 E4220006 E4220007 ;
ARRAY E42900 E4290001 E4290002 E4290003 E4290004 E4290005 E4290006 E4290007 ;
ARRAY E52400 E5240001 E5240002 E5240003 E5240004 E5240005 E5240006 E5240007 ;
ARRAY E52500 E5250001 E5250002 E5250003 E5250004 E5250005 E5250006 E5250007 ;
ARRAY E52600 E5260001 E5260002 E5260003 E5260004 E5260005 E5260006 E5260007 ;
ARRAY E53400 E5340001 E5340002 E5340003 E5340004 E5340005 E5340006 E5340007 ;
ARRAY E54200 E5420001 E5420002 E5420003 E5420004 E5420005 E5420006 E5420007 ;
ARRAY E54300 E5430001 E5430002 E5430003 E5430004 E5430005 E5430006 E5430007 ;
ARRAY E54600 E5460001 E5460002 E5460003 E5460004 E5460005 E5460006 E5460007 ;
ARRAY E55100 E5510001 E5510002 E5510003 E5510004 E5510005 E5510006 E5510007 ;
ARRAY E55200 E5520001 E5520002 E5520003 E5520004 E5520005 E5520006 E5520007 ;
ARRAY E59800 E5980001 E5980002 E5980003 E5980004 E5980005 E5980006 E5980007 ;
ARRAY E62000 E6200001 E6200002 E6200003 E6200004 E6200005 E6200006 E6200007 ;
ARRAY E65700 E6570001 E6570002 E6570003 E6570004 E6570005 E6570006 E6570007 ;
ARRAY E65800 E6580001 E6580002 E6580003 E6580004 E6580005 E6580006 E6580007 ;
ARRAY E66000 E6600001 E6600002 E6600003 E6600004 E6600005 E6600006 E6600007 ;
ARRAY E66700 E6670001 E6670002 E6670003 E6670004 E6670005 E6670006 E6670007 ;
ARRAY E76200 E7620001 E7620002 E7620003 E7620004 E7620005 E7620006 E7620007 ;
ARRAY E76300 E7630001 E7630002 E7630003 E7630004 E7630005 E7630006 E7630007 ;
ARRAY E76400 E7640001 E7640002 E7640003 E7640004 E7640005 E7640006 E7640007 ;
ARRAY E77200 E7720001 E7720002 E7720003 E7720004 E7720005 E7720006 E7720007 ;
ARRAY E78000 E7800001 E7800002 E7800003 E7800004 E7800005 E7800006 E7800007 ;
ARRAY E78100 E7810001 E7810002 E7810003 E7810004 E7810005 E7810006 E7810007 ;
ARRAY E78400 E7840001 E7840002 E7840003 E7840004 E7840005 E7840006 E7840007 ;
ARRAY E78900 E7890001 E7890002 E7890003 E7890004 E7890005 E7890006 E7890007 ;
ARRAY E79000 E7900001 E7900002 E7900003 E7900004 E7900005 E7900006 E7900007 ;
ARRAY E82200 E8220001 E8220002 E8220003 E8220004 E8220005 E8220006 E8220007 ;
ARRAY E83100 E8310001 E8310002 E8310003 E8310004 E8310005 E8310006 E8310007 ;
ARRAY E86800 E8680001 E8680002 E8680003 E8680004 E8680005 E8680006 E8680007 ;
ARRAY E86900 E8690001 E8690002 E8690003 E8690004 E8690005 E8690006 E8690007 ;
ARRAY E87100 E8710001 E8710002 E8710003 E8710004 E8710005 E8710006 E8710007 ;
ARRAY E87800 E8780001 E8780002 E8780003 E8780004 E8780005 E8780006 E8780007 ;
ARRAY E97300 E9730001 E9730002 E9730003 E9730004 E9730005 E9730006 E9730007 ;
ARRAY E97400 E9740001 E9740002 E9740003 E9740004 E9740005 E9740006 E9740007 ;
ARRAY E97500 E9750001 E9750002 E9750003 E9750004 E9750005 E9750006 E9750007 ;
ARRAY E98300 E9830001 E9830002 E9830003 E9830004 E9830005 E9830006 E9830007 ;
ARRAY E99200 E9920001 E9920002 E9920003 E9920004 E9920005 E9920006 E9920007 ;

Appendix 2: Employment Variable Creation

```
ARRAY E99500 E9950001 E9950002 E9950003 E9950004 E9950005 E9950006 E9950007 ;  
ARRAY E100e E100e01 E100e02 E100e03 E100e04 E100e05 E100e06 E100e07;  
ARRAY E1001e E1001e01 E1001e02 E1001e03 E1001e04 E1001e05 E1001e06 E1001e07;  
ARRAY E26800[7] E2680001-E2680007;  
ARRAY E26900[7] E2690001-E2690007;  
ARRAY E29200[7] E2920001-E2920007;  
ARRAY E48200[7] E4820001-E4820007;  
ARRAY E72000[7] E7200001-E7200007;  
ARRAY E93100[7] E9310001-E9310007;
```

```
*****SECTION 1: START WAGES REPORTED HOURLY, WEEKLY, BIWEEKLY, MONTHLY, AND  
ANNUALLY****/
```

```
/*Part A—Hourly: Set up the hourly wage for youths who report an hourly wage*/
```

```
ARRAY HRWAGE HRWAGE01 HRWAGE02 HRWAGE03 HRWAGE04 HRWAGE05 HRWAGE06  
HRWAGE07;  
DO I=1 TO 7;  
HRWAGE[I]=0;
```

```
/*under 16, start wage, no overtime*/
```

```
IF (E17100[I]=1 AND E37800[I]=0) then DO;  
  IF (E22900[I]>0) then HRWAGE[I]=E22900[I];  
  IF (E23000[I]>0) then HRWAGE[I]=E23000[I];  
  IF (E22900[I]=-3 or E23000[I]=-3) then HRWAGE[I]=E23200[I];  
  IF (E22900[I]=-2 or E23000[I]=-2) then HRWAGE[I]=E23200[I];  
  IF (E22900[I]=-1 or E23000[I]=-1) then HRWAGE[I]=-1;  
  IF E23900[I]>0 then HRWAGE[I]=E23900[I];
```

```
END;
```

```
/*16+, start wage, no overtime*/
```

```
IF (E17100[I]=0 AND E79200[I]=0) then DO;  
  IF (E65700[I]>0) then HRWAGE[I]=E65700[I];  
  IF (E65800[I]>0) then HRWAGE[I]=E65800[I];  
  IF (E65700[I]=-3 or E65800[I]=-3) then HRWAGE[I]=E66000[I];  
  IF (E65700[I]=-2 or E65800[I]=-2) then HRWAGE[I]=E66000[I];  
  IF (E65700[I]=-1 or E65800[I]=-1) then HRWAGE[I]=-1;  
  IF E66700[I]>0 then HRWAGE[I]=E66700[I];
```

```
END;
```

```
END;
```

```
*****Part B—Weekly: Set up the hourly wage for youths who report a weekly, daily, or other time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in weekly units.***/
```

```
ARRAY WEEKLY WEEKLY01 WEEKLY02 WEEKLY03 WEEKLY04 WEEKLY05 WEEKLY06 WEEKLY07;  
DO I=1 TO 7;  
WEEKLY[I]=0;
```

```
/*Under 16, weekly start wage divided by the number of hours worked per week*/
```

```
/*no overtime*/
```

```
IF (E17100[I]=1 AND E37800[I]=0) then DO;  
  IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E33400[I]>0  
    and E19000[I]>0 then WEEKLY[I]=(E33400[I]/E19000[I]);
```

```
/*estimated*/
```

```
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E33600[I]>0  
  and E19000[I]>0 and E33400[I]>-3 then WEEKLY[I]=(E33600[I]/E19000[I]);
```

```
/*interviewer corrected*/
```

Appendix 2: Employment Variable Creation

```
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E34400[I]>0  
    and E19000[I]>0 and E33400[I]>-3 then WEEKLY[I]=(E34400[I]/E19000[I]);  
/*respondent corrected*/  
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E35300[I]>0  
    and E19000[I]>0 and E33400[I]>-3 then WEEKLY[I]=(E35300[I]/E19000[I]);  
/*missing values*/  
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and (E19000[I]=-1  
    or E19000[I]=-2 or E19000[I]=-3) then WEEKLY[I]=E19000[I];  
IF (E19000[I]=0) then WEEKLY[I]=-6;  
END;  
  
/*16+, weekly start wage divided by the number of hours worked per week*/  
/*no overtime*/  
ELSE IF E17100[I]=0 AND E79200[I]=0 then DO;  
    IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E76200[I]>0  
        and E59800[I]>0 then WEEKLY[I]=(E76200[I]/E59800[I]);  
    /*estimated*/  
    IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E76400[I]>0  
        and E59800[I]>0 and E76200[I]>-3 then WEEKLY[I]=(E76400[I]/E59800[I]);  
    /*interviewer corrected*/  
    IF (E62000[I]=2 | E38200[I]=3 | E62000[I]=7 | E38200[I] GE 9) and E77200[I]>0  
        and E59800[I]>0 and E76200[I]>-3 then WEEKLY[I]=(E77200[I]/E59800[I]);  
    /*respondent corrected*/  
    IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E78100[I]>0  
        and E59800[I]>0 and E76200[I]>-3 then WEEKLY[I]=(E78100[I]/E59800[I]);  
    /*missing values*/  
    IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and (E59800[I]=-1  
        or E59800[I]=-2 or E59800[I]=-3) then WEEKLY[I]=E59800[I];  
    IF (E62000[I]=2 | E62000[I]=3 | E62000[I] GE 9) and E59800[I]=0 then WEEKLY[I]=-6;  
END;  
END;
```

*****Part C—Biweekly: Set up the hourly wage for youths who report a biweekly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in biweekly units. */

```
ARRAY BIWKLY BIWKLY01 BIWKLY02 BIWKLY03 BIWKLY04 BIWKLY05 BIWKLY06 BIWKLY07;  
DO I=1 TO 7;  
BIWKLY[I]=0;  
  
/*under 16, bi-wkly start wage divided by 2 times the number of hours worked per week */  
/*no overtime*/  
IF E17100[I]=1 AND E37800[I]=0 then DO;  
    IF (E19200[I]=4) and E33400[I]>0 and E19000[I]>0 then BIWKLY[I]=(E33400[I]/(2*E19000[I]));  
    /*estimated*/  
    IF (E19200[I]=4) and E33600[I]>0 and E19000[I]>0 and E33400[I]>-3  
        then BIWKLY[I]=(E33600[I]/(2*E19000[I]));  
    /*interviewer corrected*/  
    IF (E19200[I]=4) and E34400[I]>0 and E19000[I]>0 and E33400[I]>-3  
        then BIWKLY[I]=(E34400[I]/(2*E19000[I]));  
    /*respondent corrected*/  
    IF (E19200[I]=4) and E35300[I]>0 and E19000[I]>0 and E33400[I]>-3  
        then BIWKLY[I]=(E35300[I]/(2*E19000[I]));  
    /*missing values*/  
    IF (E19200[I]=4) and (E19000[I]=-1 or E19000[I]=-2 or E19000[I]=-3) then BIWKLY[I]=E19000[I];  
    IF (E19200[I]=4) and (E19000[I]=0) then BIWKLY[I]=-6;  
END;
```

Appendix 2: Employment Variable Creation

```
/*16+, bi-wkly start wage divided by 2 times the number of hours worked per week*/
/*no overtime*/
    ELSE IF E17100[I]=0 AND E79200[I]=0 then DO;
        IF (E62000[I]=4) and E76200[I]>0 and E59800[I]>0 then BIWKLY[I]=(E76200[I]/(2*E59800[I]));
    /*estimated*/
        IF (E62000[I]=4) and E76400[I]>0 and E59800[I]>0 and E76200[I]>-3
            then BIWKLY[I]=(E76400[I]/(2*E59800[I]));
    /*interviewer corrected*/
        IF (E62000[I]=4) and E77200[I]>0 and E59800[I]>0 and E76200[I]>-3
            then BIWKLY[I]=(E77200[I]/(2*E59800[I]));
    /*respondent corrected*/
        IF (E62000[I]=4) and E78100[I]>0 and E59800[I]>0 and E76200[I]>-3
            then BIWKLY[I]=(E78100[I]/(2*E59800[I]));
    /*missing values*/
        IF (E62000[I]=4) and (E59800[I]=-1 or E59800[I]=-2 or E59800[I]=-3) then BIWKLY[I]=E59800[I];
        IF (E62000[I]=4) and E59800[I]=0 then BIWKLY[I]=-6;
    END;
END;
```

*****Part D—Monthly: Set up the hourly wage for youths who report a monthly or semi-monthly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in monthly units.*/

```
ARRAY MONTH MONTH01 MONTH02 MONTH03 MONTH04 MONTH05 MONTH06 MONTH07;
DO I=1 TO 7;
MONTH[I]=0;
```

```
/*under 16, month start wage divided by 4.3 times the number of hours worked per week*/
/*no overtime*/
    IF E17100[I]=1 AND E37800[I]=0 then DO;
        IF (E19200[I]=5 | E19200[I]=8) and E33400[I]>0 and E19000[I]>0
            then MONTH[I]=(E33400[I]/(4.3*E19000[I]));
    /*estimated*/
        IF (E19200[I]=5 | E19200[I]=8) and E33600[I]>0 and E19000[I]>0 and E33400[I]>-3
            then MONTH[I]=(E33600[I]/(4.3*E19000[I]));
    /*interviewer corrected*/
        IF (E19200[I]=5 | E19200[I]=8) and E34400[I]>0 and E19000[I]>0 and E33400[I]>-3
            then MONTH[I]=(E34400[I]/(4.3*E19000[I]));
    /*respondent corrected*/
        IF (E19200[I]=5 | E19200[I]=8) and E35300[I]>0 and E19000[I]>0 and E33400[I]>-3
            then MONTH[I]=(E35300[I]/(4.3*E19000[I]));
    /*missing values*/
        IF (E19200[I]=5 | E19200[I]=8) and (E19000[I]=-1 or E19000[I]=-2 or E19000[I]=-3)
            then MONTH[I]=E19000[I];
        IF (E19200[I]=5 | E19200[I]=8) and E19000[I]=0 then MONTH[I]=-6;
    END;
```

```
/*16+, month start wage divided by 4.3 times the number of hours worked per week*/
/*no overtime*/
    ELSE IF E17100[I]=0 AND E79200[I]=0 then DO;
        IF (E62000[I]=5 | E62000[I]=8) and E76200[I]>0 and E59800[I]>0
            then MONTH[I]=(E76200[I]/(4.3*E59800[I]));
    /*estimated*/
        IF (E62000[I]=5 | E62000[I]=8) and E76400[I]>0 and E59800[I]>0 and E76200[I]>-3
            then MONTH[I]=(E76400[I]/(4.3*E59800[I]));
    /*interviewer corrected*/
        IF (E62000[I]=5 | E38200[I]=8) and E77200[I]>0 and E59800[I]>0 and E76200[I]>-3
            then MONTH[I]=(E77200[I]/(4.3*E59800[I]));
```

```

/*respondent corrected*/
IF (E62000[I]=5 | E62000[I]=8) and E78100[I]>0 and E59800[I]>0 and E76200[I]>-3
    then MONTH[I]=(E78100[I]/(4.3*E59800[I]));
/*missing values*/
IF (E62000[I]=5 | E62000[I]=8) and (E59800[I]=-1 or E59800[I]=-2 or E59800[I]=-3)
    then MONTH[I]=E59800[I];
IF (E62000[I]=5 | E62000[I]=8) and E59800[I]=0 then MONTH[I]=-6;
END;
END;

*****Part E—Annual: Set up the hourly wage for youths who report an annual time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in annual units.*/

ARRAY ANNUAL ANNUAL01 ANNUAL02 ANNUAL03 ANNUAL04 ANNUAL05 ANNUAL06 ANNUAL07;
DO I=1 TO 7;
ANNUAL[I]=0;

/*under 16, annual start wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
IF E17100[I]=1 AND E37800[I]=0 then DO;
IF E19200[I]=6 and E19000[I]>0 and E33400[I]>0 and E35600[I]>0
    then ANNUAL[I]=(E33400[I]/(E35600[I]*E19000[I]));
/*estimated*/
IF (E19200[I]=6) and E33600[I]>0 and E19000[I]>0 and E33400[I]>-3
    then ANNUAL[I]=(E33600[I]/(E35600[I]*E19000[I]));
/*interviewer corrected*/
IF (E19200[I]=6) and E34400[I]>0 and E19000[I]>0 and E33400[I]>-3
    then ANNUAL[I]=(E34400[I]/(E35600[I]*E19000[I]));
/*respondent corrected*/
IF (E19200[I]=6) and E35300[I]>0 and E19000[I]>0 and E33400[I]>-3
    then ANNUAL[I]=(E35300[I]/(E35600[I]*E19000[I]));
/*missing values*/
IF E19200[I]=6 and (E19000[I]=-1 or E19000[I]=-2 or E19000[I]=-3) then ANNUAL[I]=E19000[I];
IF E19200[I]=6 and (E35600[I]=-1 or E35600[I]=-2 or E35600[I]=-3) then ANNUAL[I]=E35600[I];
IF E19200[I]=6 and E19000[I]=0 then ANNUAL[I]=-6;
END;

/*16+, annual start wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
ELSE IF E17100[I]=0 AND E79200[I]=0 then DO;
IF E62000[I]=6 and E76200[I]>0 and E78400[I]>0 and E59800[I]>0
    then ANNUAL[I]=(E76200[I]/(E78400[I]*E59800[I]));
/*estimated*/
IF (E62000[I]=6) and E76400[I]>0 and E59800[I]>0 and E76200[I]>-3
    then ANNUAL[I]=(E76400[I]/(E78400[I]*E59800[I]));
/*interviewer corrected*/
IF (E62000[I]=6) and E77200[I]>0 and E59800[I]>0 and E76200[I]>-3
    then ANNUAL[I]=(E77200[I]/(E78400[I]*E59800[I]));
/*respondent corrected*/
IF (E62000[I]=6) and E78100[I]>0 and E59800[I]>0 and E76200[I]>-3
    then ANNUAL[I]=(E78100[I]/(E78400[I]*E59800[I]));
/*missing values*/
IF E62000[I]=6 and (E59800[I]=-1 or E59800[I]=-2 or E59800[I]=-3) then ANNUAL[I]=E59800[I];
IF E62000[I]=6 and (E78400[I]=-1 or E78400[I]=-2 or E78400[I]=-3) then ANNUAL[I]=E78400[I];
IF E62000[I]=6 and E59800[I]=0 then ANNUAL[I]=-6;
END;
END;

```

*******Part F—Other:** Set up the hourly wage for youths who report a non-hourly time unit. Both the hourly wage reported in the non-overtime and the overtime loops can be used without any further calculations.*/

```
ARRAY OTHER OTHER01 OTHER02 OTHER03 OTHER04 OTHER05 OTHER06 OTHER07;
DO I=1 TO 7;
OTHER[I]=0;
```

```
/*under 16, start wage*/
IF E17100[I]=1 AND E37800[I]=0 then DO;
  IF (E36100[I]>0) then OTHER[I]=E36100[I];
  IF (E36200[I]>0) then OTHER[I]=E36200[I];
END;

/*16+, start wage*/
ELSE IF E17100[I]=0 AND E79200[I]=0 then DO;
  IF (E78900[I]>0) then OTHER[I]=E78900[I];
  IF (E79000[I]>0) then OTHER[I]=E79000[I];
END;
END;
```

*****Create the Hourly Rate of Pay*****

```
ARRAY HRWG HRWG01 HRWG02 HRWG03 HRWG04 HRWG05 HRWG06 HRWG07;
DO I=1 TO 7;
HRWG[I]=0;
```

```
IF HRWAGE[I] EQ -4 OR WEEKLY[I] EQ -4 OR BIWKLY[I] EQ -4 OR MONTH[I] EQ -4 OR ANNUAL[I]
  EQ -4 OR OTHER[I] EQ -4 THEN HRWG[I]=-4;
IF HRWAGE[I] EQ -3 OR WEEKLY[I] EQ -3 OR BIWKLY[I] EQ -3 OR MONTH[I] EQ -3 OR ANNUAL[I]
  EQ -3 OR OTHER[I] EQ -3 THEN HRWG[I]=-3;
IF HRWAGE[I] EQ -2 OR WEEKLY[I] EQ -2 OR BIWKLY[I] EQ -2 OR MONTH[I] EQ -2 OR ANNUAL[I]
  EQ -2 OR OTHER[I] EQ -2 THEN HRWG[I]=-2;
IF HRWAGE[I] EQ -1 OR WEEKLY[I] EQ -1 OR BIWKLY[I] EQ -1 OR MONTH[I] EQ -1 OR ANNUAL[I]
  EQ -1 OR OTHER[I] EQ -1 THEN HRWG[I]=-1;
```

/* If R refuses the first time asked, R is jumped out of wage section*/
IF E33400[I]=-1 OR E76200[I]=-1 then HRWG[I]=-1;

/*If R refuses or doesn't know the estimated wage, R is jumped out of the wage section*/
IF E33600[I]=-1 OR E76400[I]=-1 THEN HRWG[I]=-1;
IF E33600[I]=-2 OR E76400[I]=-2 THEN HRWG[I]=-2;

```
IF ANNUAL[I] GT 0 THEN HRWG[I]=ANNUAL[I];  IF MONTH[I] GT 0 THEN HRWG[I]=MONTH[I];
IF BIWKLY[I] GT 0 THEN HRWG[I]=BIWKLY[I];  IF WEEKLY[I] GT 0 THEN HRWG[I]=WEEKLY[I];
IF HRWAGE[I] GT 0 THEN HRWG[I]=HRWAGE[I];  IF OTHER[I] GT 0 THEN HRWG[I]=OTHER[I];
END;
```

*****SECTION 2: STOP DATE WAGES FOR YOUTH*****

***Part A—Hourly:** Set up the hourly wage for youths who report an hourly wage*/

```
DO I=1 TO 7;
/*under 16, end wage, no overtime */
  IF (E17100[I]=1 AND E37800[I]=1) then DO;
    IF (E41900[I]>0) then HRWAGE[I]=E41900[I];
```

```

IF (E42000[I]>0) then HRWAGE[I]=E42000[I];
IF (E41900[I]=-3 or E42000[I]=-3) then HRWAGE[I]=E42200[I];
IF (E41900[I]=-2 or E42000[I]=-2) then HRWAGE[I]=E42200[I];
IF (E41900[I]=-1 or E42000[I]=-1) then HRWAGE[I]=-1;
IF E42900[I]>0 then HRWAGE[I]=E42900[I];
END;

/*16+, end wage, no overtime */
ELSE IF (E17100[I]=0 AND E79200[I]=1) then DO;
  IF (E86800[I]>0) then HRWAGE[I]=E86800[I];
  IF (E86900[I]>0) then HRWAGE[I]=E86900[I];
  IF (E86800[I]=-3 or E86900[I]=-3) then HRWAGE[I]=E87100[I];
  IF (E86800[I]=-2 or E86900[I]=-2) then HRWAGE[I]=E87100[I];
  IF (E86800[I]=-1 or E86900[I]=-1) then HRWAGE[I]=-1;
  IF E87800[I]>0 then HRWAGE[I]=E87800[I];
END;
END;

```

*******Part B—Weekly:** Set up the hourly wage for youths who report a weekly, daily, or other time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in weekly units.*/

```

DO I=1 TO 7;
/*under 16, weekly end wage divided by the number of hours worked per week*/
/*no overtime*/
  IF (E17100[I]=1 AND E37800[I]=1) then DO;
    IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E52400[I]>0 and E37900[I]>0
      then WEEKLY[I]=(E52400[I]/E37900[I]);
  /*estimated*/
    IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E52600[I]>0
      and E37900[I]>0 and E52400[I]>-3 then WEEKLY[I]=(E52600[I]/E37900[I]);
  /*interviewer corrected*/
    IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E53400[I]>0
      and E37900[I]>0 and E52400[I]>-3 then WEEKLY[I]=(E53400[I]/E37900[I]);
  /*respondent corrected*/
    IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E54300[I]>0
      and E37900[I]>0 and E52400[I]>-3 then WEEKLY[I]=(E54300[I]/E37900[I]);
  /*missing values*/
    IF (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3) then WEEKLY[I]=E37900[I];
    IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E37900[I]=0 then WEEKLY[I]=0;
  END;

```

```

/*16+, weekly end wage divided by the number of hours worked per week*/
/*no overtime*/
ELSE IF E17100[I]=0 AND E79200[I]=1 then DO;
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E97300[I]>0 and E82200[I]>0
    then WEEKLY[I]=(E97300[I]/E82200[I]);
  /*estimated*/
    IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E97500[I]>0
      and E82200[I]>0 and E97300[I]>-3 then WEEKLY[I]=(E97500[I]/E82200[I]);
  /*interviewer corrected*/
    IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E98300[I]>0
      and E82200[I]>0 and E97300[I]>-3 then WEEKLY[I]=(E98300[I]/E82200[I]);
  /*respondent corrected*/
    IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E99200[I]>0
      and E82200[I]>0 and E97300[I]>-3 then WEEKLY[I]=(E99200[I]/E82200[I]);
  /*missing values*/
    IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and (E82200[I]=-1

```

Appendix 2: Employment Variable Creation

```
        or E82200[I]=-2 or E82200[I]=-3) then WEEKLY[I]=E82200[I];
        IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E82200[I]=0 then WEEKLY[I]=-6;
    END;
END;
```

*****Part C—Biweekly: Set up the hourly wage for youths who report a biweekly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in biweekly units.*/

```
DO I=1 TO 7;
/*Under 16, bi-wkly end wage divided by 2 times the number of hours worked per week*/
/*no overtime*/
    IF E17100[I]=1 AND E37800[I]=1 then DO;
        IF (E38200[I]=4) and E52400[I]>0 and E37900[I]>0 then BIWKLY[I]=(E52400[I]/(2*E37900[I]));
    /*estimated*/
        IF (E38200[I]=4) and E52600[I]>0 and E37900[I]>0 and E52400[I]>-3
            then BIWKLY[I]=(E52600[I]/(2*E37900[I]));
    /*interviewer corrected*/
        IF (E38200[I]=4) and E53400[I]>0 and E37900[I]>0 and E52400[I]>-3
            then BIWKLY[I]=(E53400[I]/(2*E37900[I]));
    /*respondent corrected*/
        IF (E38200[I]=4) and E54300[I]>0 and E37900[I]>0 and E52400[I]>-3
            then BIWKLY[I]=(E54300[I]/(2*E37900[I]));
    /*missing values*/
        IF (E38200[I]=4) and (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3) then BIWKLY[I]=E37900[I];
        IF (E38200[I]=4) and E37900[I]=0 then BIWKLY[I]=-6;
    END;
```

/*16+, bi-wkly end wage divided by 2 times the number of hours worked per week*/

```
/*no overtime*/
    ELSE IF E17100[I]=0 AND E79200[I]=1 then DO;
        IF (E83100[I]=4) and E97300[I]>0 and E82200[I]>0 then BIWKLY[I]=(E97300[I]/(2*E82200[I]));
    /*estimated*/
        IF (E83100[I]=4) and E97500[I]>0 and E82200[I]>0 and E97300[I]>-3
            then BIWKLY[I]=(E97500[I]/(2*E82200[I]));
    /*interviewer corrected*/
        IF (E83100[I]=4) and E98300[I]>0 and E82200[I]>0 and E97300[I]>-3
            then BIWKLY[I]=(E98300[I]/(2*E82200[I]));
    /*respondent corrected*/
        IF (E83100[I]=4) and E99200[I]>0 and E82200[I]>0 and E97300[I]>-3
            then BIWKLY[I]=(E99200[I]/(2*E82200[I]));
    /*missing values*/
        IF (E83100[I]=4) and (E82200[I]=-1 or E82200[I]=-2 or E82200[I]=-3) then BIWKLY[I]=E82200[I];
        IF (E83100[I]=4) and E82200[I]=0 then BIWKLY[I]=-6;
    END;
END;
```

*****Part D—Monthly: Set up the hourly wage for youths who report a month, or semi-monthly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in month units.*/

```
DO I=1 TO 7;
/*under 16, month end wage divided by 4.3 times the number of hours worked per week*/
/*no overtime*/
    IF E17100[I]=1 AND E37800[I]=1 then DO;
        IF (E38200[I]=5 | E38200[I]=8) and E52400[I]>0 and E37900[I]>0
            then MONTH[I]=(E52400[I]/(4.3*E37900[I]));
    /*estimated*/
        IF (E38200[I]=5 | E38200[I]=8) and E52600[I]>0 and E37900[I]>0 and E52400[I]>-3
```

Appendix 2: Employment Variable Creation

```
        then MONTH[I]=(E52600[I]/(4.3*E37900[I]));  
/*interviewer corrected*/  
        IF (E38200[I]=5 | E38200[I]=8) and E53400[I]>0 and E37900[I]>0 and E52400[I]>-3  
            then MONTH[I]=(E53400[I]/(4.3*E37900[I]));  
/*respondent corrected*/  
        IF (E38200[I]=5 | E38200[I]=8) and E54300[I]>0 and E37900[I]>0 and E52400[I]>-3  
            then MONTH[I]=(E54300[I]/(4.3*E37900[I]));  
/*missing values*/  
        IF (E38200[I]=5 | E38200[I]=8) and (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3)  
            then MONTH[I]=E37900[I];  
        IF (E38200[I]=5 | E38200[I]=8) and E37900[I]=0 then MONTH[I]=-6;  
    END;  
  
/*16+, month end wage divided by 4.3 times the number of hours worked per week*/  
/*no overtime*/  
    ELSE IF E17100[I]=0 AND E79200[I]=1 then DO;  
        IF (E83100[I]=5 | E83100[I]=8) and E97300[I]>0 and E82200[I]>0  
            then MONTH[I]=(E97300[I]/(4.3*E82200[I]));  
/*estimated*/  
        IF (E83100[I]=5 | E83100[I]=8) and E97500[I]>0 and E82200[I]>0 and E97300[I]>-3  
            then MONTH[I]=(E97500[I]/(4.3*E82200[I]));  
/*interviewer corrected*/  
        IF (E83100[I]=5 | E83100[I]=8) and E98300[I]>0 and E82200[I]>0 and E97300[I]>-3  
            then MONTH[I]=(E98300[I]/(4.3*E82200[I]));  
/*respondent corrected*/  
        IF (E83100[I]=5 | E83100[I]=8) and E99200[I]>0 and E82200[I]>0 and E97300[I]>-3  
            then MONTH[I]=(E99200[I]/(4.3*E82200[I]));  
/*missing values*/  
        IF (E83100[I]=5 | E83100[I]=8) and (E82200[I]=-1 or E82200[I]=-2 or E82200[I]=-3)  
            then MONTH[I]=E82200[I];  
        IF (E83100[I]=5 | E83100[I]=8) and E82200[I]=0 then MONTH[I]=-6;  
    END;  
END;
```

*****Part E—Annual: Set up the hourly wage for youths who report an annual time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in annual units.*/

```
DO I=1 TO 7;  
/*Under 16, annual end wage divided by the # of weeks per year paid for times the # of hours worked per week*/  
/*no overtime*/  
    IF E17100[I]=1 AND E37800[I]=1 then DO;  
        IF E38200[I]=6 and E52400[I]>0 and E54600[I]>0 and E37900[I]>0  
            then ANNUAL[I]=(E52400[I]/(E54600[I]*E37900[I]));  
/*estimated*/  
        IF (E38200[I]=6) and E52600[I]>0 and E37900[I]>0 and E52400[I]>-3  
            then ANNUAL[I]=(E52600[I]/(E54600[I]*E37900[I]));  
/*interviewer corrected*/  
        IF (E38200[I]=6) and E53400[I]>0 and E37900[I]>0 and E52400[I]>-3  
            then ANNUAL[I]=(E53400[I]/(E54600[I]*E37900[I]));  
/*respondent corrected*/  
        IF (E38200[I]=6) and E54300[I]>0 and E37900[I]>0 and E52400[I]>-3  
            then ANNUAL[I]=(E54300[I]/(E54600[I]*E37900[I]));  
/*missing values*/  
        IF E38200[I]=6 and (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3) then ANNUAL[I]=E37900[I];  
        IF E38200[I]=6 and (E54600[I]=-1 or E54600[I]=-2 or E54600[I]=-3) then ANNUAL[I]=E54600[I];  
        IF E38200[I]=6 and E37900[I]=0 then ANNUAL[I]=-6;  
    END;
```

```

/*16+, annual end wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
    ELSE IF E17100[I]=0 AND E79200[I]=1 then DO;
        IF E83100[I]=6 and E97300[I]>0 and E99500[I]>0 and E82200[I]>0
            then ANNUAL[I]=(E97300[I]/(E99500[I]*E82200[I]));
    /*estimated*/
        IF (E83100[I]=6) and E97500[I]>0 and E82200[I]>0 and E97300[I]>-3
            then ANNUAL[I]=(E97500[I]/(E99500[I]*E82200[I]));
    /*interviewer corrected*/
        IF (E83100[I]=6) and E98300[I]>0 and E82200[I]>0 and E97300[I]>-3
            then ANNUAL[I]=(E98300[I]/(E99500[I]*E82200[I]));
    /*respondent corrected*/
        IF (E83100[I]=6) and E99200[I]>0 and E82200[I]>0 and E97300[I]>-3
            then ANNUAL[I]=(E99200[I]/(E99500[I]*E82200[I]));
    /*missing values*/
        IF E83100[I]=6 and (E82200[I]=-1 or E82200[I]=-2 or E82200[I]=-3) then ANNUAL[I]=E82200[I];
        IF E83100[I]=6 and (E99500[I]=-1 or E99500[I]=-2 or E99500[I]=-3) then ANNUAL[I]=E99500[I];
        IF E83100[I]=6 and E82200[I]=0 then ANNUAL[I]=-6;
    END;
END;

```

*******Part F—Other:** Set up the hourly wage for youths who report a non-hourly time unit. Both the hourly wage reported in the non-overtime and the overtime loops can be used without any further calculations.*/

```

DO I=1 TO 7;
/*under 16, end wage*/
    IF E17100[I]=1 AND E37800[I]=1 then DO;
        IF (E55100[I]>0) then OTHER[I]=E55100[I];
        IF (E55200[I]>0) then OTHER[I]=E55200[I];
    END;

```

```

/*16+, end wage*/
    ELSE IF E17100[I]=0 AND E79200[I]=1 then DO;
        IF (E100e[I]>0) then OTHER[I]=E100e[I];
        IF (E1001e[I]>0) then OTHER[I]=E1001e[I];
    END;
END;

```

*****Create the Hourly Rate of Pay*****

```

DO I=1 TO 7;
IF HRWAGE[I] EQ -4 OR WEEKLY[I] EQ -4 OR BIWKLY[I] EQ -4 OR MONTH[I] EQ -4 OR ANNUAL[I]
    EQ -4 OR OTHER[I] EQ -4 THEN HRWG[I]=-4;
IF HRWAGE[I] EQ -3 OR WEEKLY[I] EQ -3 OR BIWKLY[I] EQ -3 OR MONTH[I] EQ -3 OR ANNUAL[I]
    EQ -3 OR OTHER[I] EQ -3 THEN HRWG[I]=-3;
IF HRWAGE[I] EQ -2 OR WEEKLY[I] EQ -2 OR BIWKLY[I] EQ -2 OR MONTH[I] EQ -2 OR ANNUAL[I]
    EQ -2 OR OTHER[I] EQ -2 THEN HRWG[I]=-2;
IF HRWAGE[I] EQ -1 OR WEEKLY[I] EQ -1 OR BIWKLY[I] EQ -1 OR MONTH[I] EQ -1 OR ANNUAL[I]
    EQ -1 OR OTHER[I] EQ -1 THEN HRWG[I]=-1;

```

/* If R refuses the first time asked, R is jumped out of wage section*/
IF E52400[I]=-1 OR E97300[I]=-1 then HRWG[I]=-1;

/*If R refuses or doesn't know the estimated wage, R is jumped out of the wage section*/
IF E52600[I]=-1 OR E97500[I]=-1 THEN HRWG[I]=-1;
IF E52600[I]=-2 OR E97500[I]=-2 THEN HRWG[I]=-2;

```
IF ANNUAL[I] GT 0 THEN HRWG[I]=ANNUAL[I];    IF MONTH[I] GT 0 THEN HRWG[I]=MONTH[I];
IF BIWKLY[I] GT 0 THEN HRWG[I]=BIWKLY[I];    IF WEEKLY[I] GT 0 THEN HRWG[I]=WEEKLY[I];
IF HRWAGE[I] GT 0 THEN HRWG[I]=HRWAGE[I];    IF OTHER[I] GT 0 THEN HRWG[I]=OTHER[I];

END;

do I=1 to 7;
  if hrwg[I]=0 and (E19200[I]=-1 or E38200[I]=-1 or E62000[I]=-1 or E83100[I]=-1) then hrwg[I]=-1;
  if hrwg[I]=0 and (E19200[I]=-2 or E38200[I]=-2 or E62000[I]=-2 or E83100[I]=-2) then hrwg[I]=-2;
  if hrwg[I]=0 and (E19200[I]=-3 or E38200[I]=-3 or E62000[I]=-3 or E83100[I]=-3) then hrwg[I]=-3;
end;

do I=1 to 7;
  if hrwg[I]=0 and (E33500[I]>-4 or E52500[I]>-4 or E76300[I]>-4 or E97400[I]>-4) then hrwg[I]=-3;
end;

do I=1 to 7; if E17100[I]=-4 then hrwg[I]=-4; end;

do I=1 to 7; if pubid=1714 then hrwg[I]=-3; end;

HRWGR01=round(hrwg01,1);      HRWGR02=round(hrwg02,1);
HRWGR03=round(hrwg03,1);      HRWGR04=round(hrwg04,1);
HRWGR05=round(hrwg05,1);      HRWGR06=round(hrwg06,1);
HRWGR07=round(hrwg07,1);

endsas;
```

HOURLY MONETARY COMPENSATION AND JOB LENGTH < 13 WEEKS INDICATOR

Variables Created: CV_HRLY_COMPENSATION
CV_JOB<13_WKS

Variables Used

This program uses all the variables listed under “Hourly Rate of Pay,” plus the following additional variables:

Name in Program	Question Name on CD	Name in Program	Question Name on CD
E2410001–04	YEMP-24100.01–04	E6960001–03	YEMP-69600.01–03
E3180001, 02	YEMP-31800.01, .02	E6970001	YEMP-69700.01
E3190001, 02	YEMP-31900.01, .02	E7460001, 02	YEMP-74600.01, .02
E4310001, 02	YEMP-43100.01, .02	E7470001, 02	YEMP-74700.01, .02
E4580001, 02	YEMP-45800.01, .02	E8800001–03	YEMP-88000.01–03
E5080001	YEMP-50800.01	E9070001, 02	YEMP-90700.01, .02
E5090001	YEMP-50900.01	E9570001, 02	YEMP-95700.01, .02
E6690001–04	YEMP-66900.01–04	E9580001, 02	YEMP-95800.01, .02

This program creates an hourly monetary compensation variable that includes information about all compensation received by the respondent, such as tips, bonuses, commissions, overtime, etc., in the calculation. Hourly monetary compensation differs from the previously described hourly rate of pay variable, which calculates only the base pay rate. This program also constructs a dummy variable indicating whether the respondent worked less than 13 weeks at a given job.

Hourly wage is constructed from stop date information for respondents who have a job lasting more than 13 weeks. For all other respondents the start wage is used. The program uses the following reporting categories:

1. The start wage for youths under the age of 16 as of the survey date who only report a start wage;
2. The stop wage for youths under the age of 16 as of the survey date who report both a start wage and a stop wage;
3. The start wage for youths age 16 or older as of the survey date who only report a start wage; and
4. The stop wage for youths age 16 or older as of the survey date who report both a start wage and a stop wage.

/* Set up the hourly wage for youths who report their wage hourly. Both the hourly wage reported in the non-overtime and the overtime loops can be used without any further calculations. */

/* NOTE: This program uses all of the arrays in the “Hourly Rate of Pay” program included above. The duplicate arrays are not reproduced here due to space considerations. Thus, this program includes the “Hourly Rate of Pay” arrays plus the following additional arrays. */

```
ARRAY E24100 E2410001 E2410002 E2410003 E2410004 E2410005 E2410006 E2410007;
ARRAY E31800 E3180001 E3180002 E3180003 E3180004 E3180005 E3180006 E3180007;
ARRAY E31900 E3190001 E3190002 E3190003 E3190004 E3190005 E3190006 E3190007;
ARRAY E24000 E2400001 E2400002 E2400003 E2400004 E2400005 E2400006 E2400007;
ARRAY E43100 E4310001 E4310002 E4310003 E4310004 E4310005 E4310006 E4310007;
ARRAY E45800 E4580001 E4580002 E4580003 E4580004 E4580005 E4580006 E4580007;
ARRAY E45900 E4590001 E4590002 E4590003 E4590004 E4590005 E4590006 E4590007;
ARRAY E50800 E5080001 E5080002 E5080003 E5080004 E5080005 E5080006 E5080007;
ARRAY E50900 E5090001 E5090002 E5090003 E5090004 E5090005 E5090006 E5090007;
ARRAY E66900 E6690001 E6690002 E6690003 E6690004 E6690005 E6690006 E6690007;
ARRAY E69600 E6960001 E6960002 E6960003 E6960004 E6960005 E6960006 E6960007;
ARRAY E69700 E6970001 E6970002 E6970003 E6970004 E6970005 E6970006 E6970007;
ARRAY E74600 E7460001 E7460002 E7460003 E7460004 E7460005 E7460006 E7460007;
ARRAY E74700 E7470001 E7470002 E7470003 E7470004 E7470005 E7470006 E7470007;
ARRAY E88000 E8800001 E8800002 E8800003 E8800004 E8800005 E8800006 E8800007;
ARRAY E90700 E9070001 E9070002 E9070003 E9070004 E9070005 E9070006 E9070007;
```

Appendix 2: Employment Variable Creation

```
ARRAY E90800 E9080001 E9080002 E9080003 E9080004 E9080005 E9080006 E9080007;
ARRAY E95700 E9570001 E9570002 E9570003 E9570004 E9570005 E9570006 E9570007;
ARRAY E95800 E9580001 E9580002 E9580003 E9580004 E9580005 E9580006 E9580007;
E99100=0;
E35200=0;
```

*******SECTION 1: START WAGES REPORTED HOURLY, WEEKLY, BIWEEKLY, MONTHLY, AND ANNUALLY****/**

/*Part A—Hourly: Set up the hourly wage for youths who report an hourly wage*/

```
ARRAY HRWAGE HRWAGE01 HRWAGE02 HRWAGE03 HRWAGE04 HRWAGE05 HRWAGE06
      HRWAGE07;
DO I=1 TO 7;
HRWAGE[I]=0;
```

/*under 16, start wage, no overtime*/

/ This segment contains the same code as the identically titled segment in Section 1, Part A—Hourly, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */*

/*under 16, start wage, overtime included */

```
IF (E17100[I]=1 AND E37800[I]=0) THEN DO;
  IF E19000[I]>0 and E29200[I]>0 THEN HRWAGE[I]=E29200[I]/E19000[I];
  IF E24100[I]>0 AND E29200[I]>0 THEN HRWAGE[I]=E29200[I]/E24100[I];
  IF E31800[I]>0 AND E19000[I]>0 THEN HRWAGE[I]=E31800[I]/E19000[I];
  IF E31800[I]>0 AND E24100[I]>0 THEN HRWAGE[I]=E31800[I]/E24100[I];
  IF E31900[I]=0 THEN HRWAGE[I]=-3;
END;
```

/*16+, start wage, no overtime*/

/ This segment contains the same code as the identically titled segment in Section 1, Part A—Hourly, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */*

/*16+, start wage, overtime included */

```
IF (E17100[I]=0 AND E79200[I]=0) THEN DO;
  IF E59800[I]>0 AND E72000[I]>0 THEN HRWAGE[I]=E72000[I]/E59800[I];
  IF E66900[I]>0 AND E72000[I]>0 THEN HRWAGE[I]=E72000[I]/E66900[I];
  IF E74600[I]>0 AND E59800[I]>0 THEN HRWAGE[I]=E74600[I]/E59800[I];
  IF E74600[I]>0 AND E66900[I]>0 THEN HRWAGE[I]=E74600[I]/E66900[I];
  IF E74700[I]=0 THEN HRWAGE[I]=-3;
END;
END;
```

*******Part B—Weekly:** Set up the hourly wage for youths who report a weekly, daily, or other time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in weekly units.*/

```
ARRAY WEEKLY WEEKLY01 WEEKLY02 WEEKLY03 WEEKLY04 WEEKLY05 WEEKLY06 WEEKLY07;
DO I=1 TO 7;
WEEKLY[I]=0;
```

/*under 16, weekly start wage divided by the number of hours worked per week*/

/*no overtime*/

/ This segment contains the same code as the identically titled segment in Section 1, Part B—Weekly, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */*

/*overtime included*/

```
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E33500[I]>0 and E19000[I]>0
  then WEEKLY[I]=(E33500[I]/E19000[I]);
```

/*estimated*/

```
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E33600[I]>0 and E19000[I]>0 and
```

Appendix 2: Employment Variable Creation

```
E33500[I]>-3 then WEEKLY[I]=(E33600[I]/E19000[I]);
/*interviewer corrected*/
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E34400[I]>0 and E19000[I]>0 and
E33500[I]>-3 then WEEKLY[I]=(E34400[I]/E19000[I]);
/*respondent corrected*/
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E35300[I]>0 and E19000[I]>0 and
E33500[I]>-3 then WEEKLY[I]=(E35200/E19000[I]);
/*missing values*/
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and (E19000[I]=-1 or E19000[I]=-2
or E19000[I]=-3) then WEEKLY[I]=E19000[I];
IF (E19200[I]=2 | E19200[I]=3 | E19200[I]=7 | E19200[I] GE 9) and E19000[I]=0 then WEEKLY[I]=-6;
END;

/*16+, weekly start wage divided by the number of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 1, Part B—Weekly, in the
“Hourly Rate of Pay” program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E76300[I]>0 and E59800[I]>0
then WEEKLY[I]=(E76300[I]/E59800[I]);
/*estimated*/
IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E76400[I]>0 and E59800[I]>0 and
E76300[I]>-3 then WEEKLY[I]=(E76400[I]/E59800[I]);
/*interviewer corrected*/
IF (E62000[I]=2 | E38200[I]=3 | E62000[I]=7 | E38200[I] GE 9) and E77200[I]>0 and E59800[I]>0 and
E76300[I]>-3 then WEEKLY[I]=(E77200[I]/E59800[I]);
/*respondent corrected*/
IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E78100[I]>0 and E59800[I]>0 and
E76300[I]>-3 then WEEKLY[I]=(E78000[I]/E59800[I]);
/*missing values*/
IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and (E59800[I]=-1 or E59800[I]=-2
or E59800[I]=-3) then WEEKLY[I]=E59800[I];
IF (E62000[I]=2 | E62000[I]=3 | E62000[I]=7 | E62000[I] GE 9) and E59800[I]=0 then WEEKLY[I]=-6;
END;
END;
```

*******Part C—Biweekly:** Set up the hourly wage for youths who report a biweekly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in biweekly units.*/

```
ARRAY BIWKLY BIWKLY01 BIWKLY02 BIWKLY03 BIWKLY04 BIWKLY05 BIWKLY06 BIWKLY07;
DO I=1 TO 7;
BIWKLY[I]=0;
```

```
/*under 16, bi-wkly start wage divided by 2 times the number of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 1, Part C—Biweekly, in the
“Hourly Rate of Pay” program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF (E19200[I]=4) and E33500[I]>0 and E19000[I]>0 then BIWKLY[I]=(E33500[I]/(2*E19000[I]));
/*estimated*/
IF (E19200[I]=4) and E33600[I]>0 and E19000[I]>0 and E33500[I]>-3
then BIWKLY[I]=(E33600[I]/(2*E19000[I]));
/*interviewer corrected*/
IF (E19200[I]=4) and E34400[I]>0 and E19000[I]>0 and E33500[I]>-3
then BIWKLY[I]=(E34400[I]/(2*E19000[I]));
/*respondent corrected*/
IF (E19200[I]=4) and E35300[I]>0 and E19000[I]>0 and E33500[I]>-3
```

Appendix 2: Employment Variable Creation

```
        then BIWKLY[I]=(E35200/(2*E19000[I]));
/*missing values*/
        IF (E19200[I]=4) and (E19000[I]=-1 or E19000[I]=-2 or E19000[I]=-3) then BIWKLY[I]=E19000[I];
        IF (E19200[I]=4) and E19000[I]=0 then BIWKLY[I]=-6;
    END;

/*16+, bi-wkly start wage divided by 2 times the number of hours worked per week*/
/*no overtime*/
    /* This segment contains the same code as the identically titled segment in Section 1, Part C—Biweekly, in the
       "Hourly Rate of Pay" program above. It is not reproduced here for space reasons. */
/*overtime included*/
    IF (E62000[I]=4) and E76300[I]>0 and E59800[I]>0 then BIWKLY[I]=(E76300[I]/(2*E59800[I]));
/*estimated*/
    IF (E62000[I]=4) and E76400[I]>0 and E59800[I]>0 and E76300[I]>-3
        then BIWKLY[I]=(E76400[I]/(2*E59800[I]));
/*interviewer corrected*/
    IF (E62000[I]=4) and E77200[I]>0 and E59800[I]>0 and E76300[I]>-3
        then BIWKLY[I]=(E77200[I]/(2*E59800[I]));
/*respondent corrected*/
    IF (E62000[I]=4) and E78100[I]>0 and E59800[I]>0 and E76300[I]>-3
        then BIWKLY[I]=(E78000[I]/(2*E59800[I]));
/*missing values*/
    IF (E62000[I]=4) and (E59800[I]=-1 or E59800[I]=-2 or E59800[I]=-3) then BIWKLY[I]=E59800[I];
    IF (E62000[I]=4) and E59800[I]=0 then BIWKLY[I]=-6;
END;
END;
```

*****Part D—Monthly: Set up the hourly wage for youths who report a monthly, or semi-monthly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in monthly units.*/

```
ARRAY MONTH MONTH01 MONTH02 MONTH03 MONTH04 MONTH05 MONTH06 MONTH07;
DO I=1 TO 7;
MONTH[I]=0;
```

```
/*under 16, month start wage divided by 4.3 times the number of hours worked per week*/
/*no overtime*/
    /* This segment contains the same code as the identically titled segment in Section 1, Part D—Monthly, in the
       "Hourly Rate of Pay" program above. It is not reproduced here for space reasons. */
/*overtime included*/
    IF (E19200[I]=5 | E19200[I]=8) and E33500[I]>0 and E19000[I]>0
        then MONTH[I]=(E33500[I]/(4.3*E19000[I]));
/*estimated*/
    IF (E19200[I]=5 | E19200[I]=8) and E33600[I]>0 and E19000[I]>0 and E33500[I]>-3
        then MONTH[I]=(E33600[I]/(4.3*E19000[I]));
/*interviewer corrected*/
    IF (E19200[I]=5 | E19200[I]=8) and E34400[I]>0 and E19000[I]>0 and E33500[I]>-3
        then MONTH[I]=(E34400[I]/(4.3*E19000[I]));
/*respondent corrected*/
    IF (E19200[I]=5 | E19200[I]=8) and E35300[I]>0 and E19000[I]>0 and E33500[I]>-3
        then MONTH[I]=(E35200/(4.3*E19000[I]));
/*missing values*/
    IF (E19200[I]=5 | E19200[I]=8) and (E19000[I]=-1 or E19000[I]=-2 or E19000[I]=-3)
        then MONTH[I]=E19000[I];
    IF (E19200[I]=5 | E19200[I]=8) and E19000[I]=0 then MONTH[I]=-6;
END;
```

/*16+, month start wage divided by 4.3 times the number of hours worked per week*/

Appendix 2: Employment Variable Creation

```
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 1, Part D—Monthly, in the
   "Hourly Rate of Pay" program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF (E62000[I]=5 | E62000[I]=8) and E76300[I]>0 and E59800[I]>0
   then MONTH[I]=(E76300[I]/(4.3*E59800[I]));
/*estimated*/
IF (E62000[I]=5 | E62000[I]=8) and E76400[I]>0 and E59800[I]>0 and E76300[I]>-3
   then MONTH[I]=(E76400[I]/(4.3*E59800[I]));
/*interviewer corrected*/
IF (E62000[I]=5 | E38200[I]=8) and E77200[I]>0 and E59800[I]>0 and E76300[I]>-3
   then MONTH[I]=(E77200[I]/(4.3*E59800[I]));
/*respondent corrected*/
IF (E62000[I]=5 | E62000[I]=8) and E78100[I]>0 and E59800[I]>0 and E76300[I]>-3
   then MONTH[I]=(E78000[I]/(4.3*E59800[I]));
/*missing values*/
IF (E62000[I]=5 | E62000[I]=8) and (E59800[I]=-1 or E59800[I]=-2 or E59800[I]=-3)
   then MONTH[I]=E59800[I];
IF (E62000[I]=5 | E62000[I]=8) and E59800[I]=0 then MONTH[I]=-6;
END;
END;
```

*******Part E—Annual:** Set up the hourly wage for youths who report an annual time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in annual units.*/

```
ARRAY ANNUAL ANNUAL01 ANNUAL02 ANNUAL03 ANNUAL04 ANNUAL05 ANNUAL06 ANNUAL07;
DO I=1 TO 7;
ANNUAL[I]=0;
```

```
/*under 16, annual start wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 1, Part E—Annual, in the
   "Hourly Rate of Pay" program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF E19200[I]=6 and E33500[I]>0 and E35600[I]>0 and E19000[I]>0
   then ANNUAL[I]=(E33500[I]/(E35600[I]*E19000[I]));
/*estimated*/
IF (E19200[I]=6) and E33600[I]>0 and E19000[I]>0 and E33500[I]>-3
   then ANNUAL[I]=(E33600[I]/(E35600[I]*E19000[I]));
/*interviewer corrected*/
IF (E19200[I]=6) and E34400[I]>0 and E19000[I]>0 and E33500[I]>-3
   then ANNUAL[I]=(E34400[I]/(E35600[I]*E19000[I]));
/*respondent corrected*/
IF (E19200[I]=6) and E35300[I]>0 and E19000[I]>0 and E33500[I]>-3
   then ANNUAL[I]=(E35200/(E35600[I]*E19000[I]));
/*missing values*/
IF E19200[I]=6 and (E19000[I]=-1 or E19000[I]=-2 or E19000[I]=-3) then ANNUAL[I]=E19000[I];
IF E19200[I]=6 and (E35600[I]=-1 or E35600[I]=-2 or E35600[I]=-3) then ANNUAL[I]=E35600[I];
IF E19200[I]=6 and E19000[I]=0 then ANNUAL[I]=0;
END;
```

```
/*16+, annual start wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 1, Part E—Annual, in the
   "Hourly Rate of Pay" program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF E62000[I]=6 and E76300[I]>0 and E78400[I]>0 and E59800[I]>0
```

```

        then ANNUAL[I]=(E76300[I]/(E78400[I]*E59800[I]));
/*estimated*/
        IF (E62000[I]=6) and E76400[I]>0 and E59800[I]>0 and E76300[I]>-3
            then ANNUAL[I]=(E76400[I]/(E78400[I]*E59800[I]));
/*interviewer corrected*/
        IF (E62000[I]=6) and E77200[I]>0 and E59800[I]>0 and E76300[I]>-3
            then ANNUAL[I]=(E77200[I]/(E78400[I]*E59800[I]));
/*respondent corrected*/
        IF (E62000[I]=6) and E78100[I]>0 and E59800[I]>0 and E76300[I]>-3
            then ANNUAL[I]=(E78000[I]/(E78400[I]*E59800[I]));
/*missing values*/
        IF E62000[I]=6 and (E59800[I]=-1 or E59800[I]=-2 or E59800[I]=-3) then ANNUAL[I]=E59800[I];
        IF E62000[I]=6 and (E78400[I]=-1 or E78400[I]=-2 or E78400[I]=-3) then ANNUAL[I]=E78400[I];
        IF E62000[I]=6 and E59800[I]=0 then ANNUAL[I]=-6;
    END;
END;

```

/***Part F—Other:** Set up the hourly wage for youths who report a non-hourly time unit. Both the hourly wage reported in the non-overtime and the overtime loops can be used without any further calculations.*/

/* This segment contains the same code as Section 1, Part F—Other, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */

*****Create the Hourly Rate of Pay*****

```

ARRAY HRCOMP HRCOMP01 HRCOMP02 HRCOMP03 HRCOMP04 HRCOMP05 HRCOMP06
      HRCOMP07;
DO I=1 TO 7;
HRCOMP[I]=0;

IF HRWAGE[I] EQ -4 OR WEEKLY[I] EQ -4 OR BIWKLY[I] EQ -4 OR MONTH[I] EQ -4 OR ANNUAL[I]
    EQ -4 OR OTHER[I] EQ -4 THEN HRCOMP[I]=-4;
IF HRWAGE[I] EQ -3 OR WEEKLY[I] EQ -3 OR BIWKLY[I] EQ -3 OR MONTH[I] EQ -3 OR ANNUAL[I]
    EQ -3 OR OTHER[I] EQ -3 THEN HRCOMP[I]=-3;
IF HRWAGE[I] EQ -2 OR WEEKLY[I] EQ -2 OR BIWKLY[I] EQ -2 OR MONTH[I] EQ -2 OR ANNUAL[I]
    EQ -2 OR OTHER[I] EQ -2 THEN HRCOMP[I]=-2;
IF HRWAGE[I] EQ -1 OR WEEKLY[I] EQ -1 OR BIWKLY[I] EQ -1 OR MONTH[I] EQ -1 OR ANNUAL[I]
    EQ -1 OR OTHER[I] EQ -1 THEN HRCOMP[I]=-1;

```

/* if R refuses the first time asked, R is jumped out of wage section*/
IF E33400[I]=-1 or E33500[I]=-1 then HRCOMP[I]=-1;
IF E76200[I]=-1 or E76300[I]=-1 then HRCOMP[I]=-1;

/*if R refuses or doesn't know the estimated wage, R is jumped out of the wage section*/
IF E33600[I]=-1 or E76400[I]=-1 then HRCOMP[I]=-1;
IF E33600[I]=-2 or E76400[I]=-2 then HRCOMP[I]=-2;

```

IF ANNUAL[I] GT 0 THEN HRCOMP[I]=ANNUAL[I];
IF MONTH[I] GT 0 THEN HRCOMP[I]=MONTH[I];
IF BIWKLY[I] GT 0 THEN HRCOMP[I]=BIWKLY[I];
IF WEEKLY[I] GT 0 THEN HRCOMP[I]=WEEKLY[I];
IF HRWAGE[I] GT 0 THEN HRCOMP[I]=HRWAGE[I];
IF OTHER[I] GT 0 THEN HRCOMP[I]=OTHER[I];
END;

```

*****SECTION 2: STOP DATE WAGES FOR YOUTH*****

Appendix 2: Employment Variable Creation

*******Part A—Hourly:** Set up the hourly wage for youths who report an hourly wage*/

DO I=1 TO 7;

/*under 16, end wage, no overtime */

/* This segment contains the same code as the identically titled segment in Section 2, Part A—Hourly, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */

/*under 16, end wage, overtime included */

IF (E17100[I]=1 AND E37800[I]=1) THEN DO;

IF E37900[I]>0 AND E48200[I]>0 THEN HRWAGE[I]=E48200[I]/E37900[I];

IF E43100[I]>0 AND E48200[I]>0 THEN HRWAGE[I]=E48200[I]/E43100[I];

IF E50800[I]>0 AND E37900[I]>0 THEN HRWAGE[I]=E50800[I]/E37900[I];

IF E50800[I]>0 AND E43100[I]>0 THEN HRWAGE[I]=E50800[I]/E43100[I];

IF E50900[I]>0 THEN HRWAGE[I]=-3;

END;

/*16+, end wage, no overtime */

/* This segment contains the same code as the identically titled segment in Section 2, Part A—Hourly, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */

/*16+, end wage, overtime included */

IF (E17100[I]=0 AND E79200[I]=1) THEN DO;

IF E82200[I]>0 AND E93100[I]>0 THEN HRWAGE[I]=E93100[I]/E82200[I];

IF E88000[I]>0 AND E93100[I]>0 THEN HRWAGE[I]=E93100[I]/E88000[I];

IF E95700[I]>0 AND E82200[I]>0 THEN HRWAGE[I]=E95700[I]/E82200[I];

IF E95700[I]>0 AND E88000[I]>0 THEN HRWAGE[I]=E95700[I]/E88000[I];

IF E95800[I]=0 THEN HRWAGE[I]=-3;

END;

END;

*******Part B—Weekly:** Set up the hourly wage for youths who report a weekly, daily, or other time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in weekly units.*/

DO I=1 TO 7;

/*under 16, weekly end wage divided by the number of hours worked per week*/

/*no overtime*/

/* This segment contains the same code as the identically titled segment in Section 2, Part B—Weekly, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */

/*overtime included*/

IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E52500[I]>0 and E37900[I]>0
then WEEKLY[I]=(E52500[I]/E37900[I]);

/*estimated*/

IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E52600[I]>0 and E37900[I]>0 and
E52500[I]>-3 then WEEKLY[I]=(E52600[I]/E37900[I]);

/*interviewer corrected*/

IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E53400[I]>0 and E37900[I]>0 and
E52500[I]>-3 then WEEKLY[I]=(E53400[I]/E37900[I]);

/*respondent corrected*/

IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E54200[I]>0 and E37900[I]>0 and
E52500[I]>-3 then WEEKLY[I]=(E54200[I]/E37900[I]);

/*missing values*/

IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and (E37900[I]=-1 or E37900[I]=-2
or E37900[I]=-3) then WEEKLY[I]=E37900[I];

IF (E38200[I]=2 | E38200[I]=3 | E38200[I]=7 | E38200[I] GE 9) and E37900[I]=0 then WEEKLY[I]=0;

END;

/*16+, weekly end wage divided by the number of hours worked per week*/

Appendix 2: Employment Variable Creation

```
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 2, Part B—Weekly, in the
   "Hourly Rate of Pay" program above. It is not reproduced here for space reasons. */
/*overtime included*/
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E97400[I]>0 and E82200[I]>0
    then WEEKLY[I]=(E97400[I]/E82200[I]);
/*estimated*/
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E97500[I]>0 and E82200[I]>0 and
    E97400[I]>-3 then WEEKLY[I]=(E97500[I]/E82200[I]);
/*interviewer corrected*/
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E98300[I]>0 and E82200[I]>0 and
    E97400[I]>-3 then WEEKLY[I]=(E98300[I]/E82200[I]);
/*respondent corrected*/
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E99100>0 and E82200[I]>0 and
    E97400[I]>-3 then WEEKLY[I]=(E99100/E82200[I]);
/*missing values*/
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and (E82200[I]=-1 or E82200[I]=-2
    or E82200[I]=-3) then WEEKLY[I]=E82200[I];
  IF (E83100[I]=2 | E83100[I]=3 | E83100[I]=7 | E83100[I] GE 9) and E82200[I]=0 then WEEKLY[I]=-6;
END;
END;
```

*****Part C—Biweekly: Set up the hourly wage for youths who report a biweekly time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in biweekly units.*/

DO I=1 TO 7;

```
/*under 16, bi-wkly end wage divided by 2 times the number of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 2, Part C—Biweekly, in the
   "Hourly Rate of Pay" program above. It is not reproduced here for space reasons. */
/*overtime included*/
  IF (E38200[I]=4) and E52500[I]>0 and E37900[I]>0 then BIWKLY[I]=(E52500[I]/(2*E37900[I]));
/*estimated*/
  IF (E38200[I]=4) and E52600[I]>0 and E37900[I]>0 and E52500[I]>-3
    then BIWKLY[I]=(E52600[I]/(2*E37900[I]));
/*interviewer corrected*/
  IF (E38200[I]=4) and E53400[I]>0 and E37900[I]>0 and E52500[I]>-3
    then BIWKLY[I]=(E53400[I]/(2*E37900[I]));
/*respondent corrected*/
  IF (E38200[I]=4) and E54200[I]>0 and E37900[I]>0 and E52500[I]>-3
    then BIWKLY[I]=(E54200[I]/(2*E37900[I]));
/*missing values*/
  IF (E38200[I]=4) and (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3) then BIWKLY[I]=E37900[I];
  IF (E38200[I]=4) and E37900[I]=0 then BIWKLY[I]=-6;
END;
```

```
/*16+, bi-wkly end wage divided by 2 times the number of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 2, Part C—Biweekly, in the
   "Hourly Rate of Pay" program above. It is not reproduced here for space reasons. */
/*overtime included*/
  IF (E83100[I]=4) and E97400[I]>0 and E82200[I]>0 then BIWKLY[I]=(E97400[I]/(2*E82200[I]));
/*estimated*/
  IF (E83100[I]=4) and E97500[I]>0 and E82200[I]>0 and E97400[I]>-3
    then BIWKLY[I]=(E97500[I]/(2*E82200[I]));
/*interviewer corrected*/
```

Appendix 2: Employment Variable Creation

```
IF (E83100[I]=4) and E98300[I]>0 and E82200[I]>0 and E97400[I]>-3  
    then BIWKLY[I]=(E98300[I]/(2*E82200[I]));  
/*respondent corrected*/  
IF (E83100[I]=4) and E99100>0 and E82200[I]>0 and E97400[I]>-3  
    then BIWKLY[I]=(E99100/(2*E82200[I]));  
/*missing values*/  
IF (E83100[I]=4) and (E82200[I]=-1 or E82200[I]=-2 or E82200[I]=-3) then BIWKLY[I]=E82200[I];  
IF (E83100[I]=4) and E82200[I]=0 then BIWKLY[I]=-6;  
END;  
END;
```

/*****Part D—Monthly: Set up the hourly wage for youths who report a month, or semi-month time unit in E-19200, E-38200, E-62000, or E-83100. All of these youths were asked to report their wage in month units.*/

```
DO I=1 TO 7;
```

```
/*under 16, month end wage divided by 4.3 times the number of hours worked per weeK*/  
/*no overtime*/  
/* This segment contains the same code as the identically titled segment in Section 2, Part D—Monthly, in the  
“Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */  
/*overtime included*/  
IF (E38200[I]=5 | E38200[I]=8) and E52500[I]>0 and E37900[I]>0  
    then MONTH[I]=(E52500[I]/(4.3*E37900[I]));  
/*estimated*/  
IF (E38200[I]=5 | E38200[I]=8) and E52600[I]>0 and E37900[I]>0 and E52500[I]>-3  
    then MONTH[I]=(E52600[I]/(4.3*E37900[I]));  
/*interviewer corrected*/  
IF (E38200[I]=5 | E38200[I]=8) and E53400[I]>0 and E37900[I]>0 and E52500[I]>-3  
    then MONTH[I]=(E53400[I]/(4.3*E37900[I]));  
/*respondent corrected*/  
IF (E38200[I]=5 | E38200[I]=8) and E54200[I]>0 and E37900[I]>0 and E52500[I]>-3  
    then MONTH[I]=(E54200[I]/(4.3*E37900[I]));  
/*missing values*/  
IF (E38200[I]=5 | E38200[I]=8) and (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3)  
    then MONTH[I]=E37900[I];  
IF (E38200[I]=5 | E38200[I]=8) and E37900[I]=0 then MONTH[I]=-6;  
END;
```

```
/*16+, month end wage divided by 4.3 times the number of hours worked per week*/
```

```
/*no overtime*/  
/* This segment contains the same code as the identically titled segment in Section 2, Part D—Monthly, in the  
“Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */  
/*overtime included*/  
IF (E83100[I]=5 | E83100[I]=8) and E97400[I]>0 and E82200[I]>0  
    then MONTH[I]=(E97400[I]/(4.3*E82200[I]));  
/*estimated*/  
IF (E83100[I]=5 | E83100[I]=8) and E97500[I]>0 and E82200[I]>0 and E97400[I]>-3  
    then MONTH[I]=(E97500[I]/(4.3*E82200[I]));  
/*interviewer corrected*/  
IF (E83100[I]=5 | E83100[I]=8) and E98300[I]>0 and E82200[I]>0 and E97400[I]>-3  
    then MONTH[I]=(E98300[I]/(4.3*E82200[I]));  
/*respondent corrected*/  
IF (E83100[I]=5 | E83100[I]=8) and E99100>0 and E82200[I]>0 and E97400[I]>-3  
    then MONTH[I]=(E99100/(4.3*E82200[I]));  
/*missing values*/  
IF (E83100[I]=5 | E83100[I]=8) and (E82200[I]=-1 or E82200[I]=-2 or E82200[I]=-3)  
    then MONTH[I]=E82200[I];
```

Appendix 2: Employment Variable Creation

```
IF (E83100[I]=5 | E83100[I]=8) and E82200[I]=0 then MONTH[I]=-6;
END;
END;

/*****Part E—Annual: Set up the hourly wage for youths who report an annual time unit in E-19200, E-38200,
E-62000, or E-83100. All of these youths were asked to report their wage in annual units.*/

DO I=1 TO 7;

/*under 16, annual end wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 2, Part E—Annual, in the
“Hourly Rate of Pay” program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF E38200[I]=6 and E52500[I]>0 and E54600[I]>0 and E37900[I]>0
then ANNUAL[I]=(E52500[I]/(E54600[I]*E37900[I]));

/*estimated*/
IF (E38200[I]=6) and E52600[I]>0 and E37900[I]>0 and E52500[I]>-3
then ANNUAL[I]=(E52600[I]/(E54600[I]*E37900[I]));

/*interviewer corrected*/
IF (E38200[I]=6) and E53400[I]>0 and E37900[I]>0 and E52500[I]>-3
then ANNUAL[I]=(E53400[I]/(E54600[I]*E37900[I]));

/*respondent corrected*/
IF (E38200[I]=6) and E54200[I]>0 and E37900[I]>0 and E52500[I]>-3
then ANNUAL[I]=(E54200[I]/(E54600[I]*E37900[I]));

/*missing values*/
IF E38200[I]=6 and (E37900[I]=-1 or E37900[I]=-2 or E37900[I]=-3) then ANNUAL[I]=E37900[I];
IF E38200[I]=6 and (E54600[I]=-1 or E54600[I]=-2 or E54600[I]=-3) then ANNUAL[I]=E54600[I];
IF E38200[I]=6 and E37900[I]=0 then ANNUAL[I]=-6;
END;

/*16+, annual end wage divided by the # of weeks per year paid for times the # of hours worked per week*/
/*no overtime*/
/* This segment contains the same code as the identically titled segment in Section 2, Part E—Annual, in the
“Hourly Rate of Pay” program above. It is not reproduced here for space reasons.*/
/*overtime included*/
IF E83100[I]=6 and E97400[I]>0 and E99500[I]>0 and E82200[I]>0
then ANNUAL[I]=(E97400[I]/(E99500[I]*E82200[I]));

/*estimated*/
IF (E83100[I]=6) and E97500[I]>0 and E82200[I]>0 and E97400[I]>-3
then ANNUAL[I]=(E97500[I]/(E99500[I]*E82200[I]));

/*interviewer corrected*/
IF (E83100[I]=6) and E98300[I]>0 and E82200[I]>0 and E97400[I]>-3
then ANNUAL[I]=(E98300[I]/(E99500[I]*E82200[I]));

/*respondent corrected*/
IF (E83100[I]=6) and E99100>0 and E82200[I]>0 and E97400[I]>-3
then ANNUAL[I]=(E99100/(E99500[I]*E82200[I]));

/*missing values*/
IF E83100[I]=6 and (E82200[I]=-1 or E82200[I]=-2 or E82200[I]=-3) then ANNUAL[I]=E82200[I];
IF E83100[I]=6 and (E99500[I]=-1 or E99500[I]=-2 or E99500[I]=-3) then ANNUAL[I]=E99500[I];
IF E83100[I]=6 and E82200[I]=0 then ANNUAL[I]=-6;
END;
END;
```

*****Part F—Other: Set up the hourly wage for youths who report a non-hourly time unit. Both the hourly wage reported in the non-overtime and the overtime loops can be used without any further calculations.*/

Appendix 2: Employment Variable Creation

/* This segment contains the same code as Section 2, Part F—Other, in the “Hourly Rate of Pay” program above. It is not reproduced here for space reasons. */

```
*****Create the Hourly Compensation*****
```

```
DO I=1 TO 7;
```

```
IF HRWAGE[I] EQ -4 OR WEEKLY[I] EQ -4 OR BIWKLY[I] EQ -4 OR MONTH[I] EQ -4 OR ANNUAL[I]  
EQ -4 OR OTHER[I] EQ -4 THEN HRCOMP[I]=-4;
```

```
IF HRWAGE[I] EQ -3 OR WEEKLY[I] EQ -3 OR BIWKLY[I] EQ -3 OR MONTH[I] EQ -3 OR ANNUAL[I]  
EQ -3 OR OTHER[I] EQ -3 THEN HRCOMP[I]=-3;
```

```
IF HRWAGE[I] EQ -2 OR WEEKLY[I] EQ -2 OR BIWKLY[I] EQ -2 OR MONTH[I] EQ -2 OR ANNUAL[I]  
EQ -2 OR OTHER[I] EQ -2 THEN HRCOMP[I]=-2;
```

```
IF HRWAGE[I] EQ -1 OR WEEKLY[I] EQ -1 OR BIWKLY[I] EQ -1 OR MONTH[I] EQ -1 OR ANNUAL[I]  
EQ -1 OR OTHER[I] EQ -1 THEN HRCOMP[I]=-1;
```

```
/* if R refuses the first time asked, R is jumped out of wage section*/
```

```
IF E52400[I]=-1 or E52500[I]=-1 then HRCOMP[I]=-1;
```

```
IF E97300[I]=-1 or E97400[I]=-1 then HRCOMP[I]=-1;
```

```
/*if R refuses or doesn't know the estimated wage, R is jumped out of the wage section*/
```

```
IF E52600[I]=-1 OR E97500[I]=-1 THEN HRCOMP[I]=-1;
```

```
IF E52600[I]=-2 OR E97500[I]=-2 THEN HRCOMP[I]=-2;
```

```
IF ANNUAL[I] GT 0 THEN HRCOMP[I]=ANNUAL[I];
```

```
IF MONTH[I] GT 0 THEN HRCOMP[I]=MONTH[I];
```

```
IF BIWKLY[I] GT 0 THEN HRCOMP[I]=BIWKLY[I];
```

```
IF WEEKLY[I] GT 0 THEN HRCOMP[I]=WEEKLY[I];
```

```
IF HRWAGE[I] GT 0 THEN HRCOMP[I]=HRWAGE[I];
```

```
IF OTHER[I] GT 0 THEN HRCOMP[I]=OTHER[I];
```

```
END;
```

```
*****SECTION 3: JOB LASTS 13 WEEKS OR LESS*****
```

```
ARRAY JLENG JLENG01 JLENG02 JLENG03 JLENG04 JLENG05 JLENG06 JLENG07;
```

```
DO I=1 TO 7;
```

```
JLENG[I]=0
```

```
JLENG[I]=-4;
```

```
IF E37800[I]>-1 OR E79200[I]>-1 then DO;
```

```
  IF E37800[I]=0 OR E79200[I]=0 then JLENG[I]=1;
```

```
  ELSE JLENG[I]=0;
```

```
END;
```

```
END;
```

```
/* to correct for those not eligible for the employment section */
```

```
do I=1 to 7;
```

```
  if E17100[I]=-4 then HRCOMP[I]=-4 and JLENG[I]=-4;
```

```
end;
```

```
/*correct for youth's being unable to report time unit rate of pay*/
```

```
do I=1 to 7;
```

```
  if HRCOMP[I]=0 and (E19200[I]=-1 or E38200[I]=-1 or E62000[I]=-1 or E83100[I]=-1) then HRCOMP[I]=-1;
```

```
  if HRCOMP[I]=0 and (E19200[I]=-2 or E38200[I]=-2 or E62000[I]=-2 or E83100[I]=-2) then HRCOMP[I]=-2;
```

```
  if HRCOMP[I]=0 and (E19200[I]=-3 or E38200[I]=-3 or E62000[I]=-3 or E83100[I]=-3) then HRCOMP[I]=-3;
```

```
end;
```

Appendix 2: Employment Variable Creation

```
/*correct for invalid skip in employment section*/
do I=1 to 7;
  if pubid=1714 then HRCOMP[I]=-3 and JLENG[I]=-3;
end;

HRCOMR01=HRCOMP01;
HRCOMR02=HRCOMP02;
HRCOMR03=HRCOMP03;
HRCOMR04=HRCOMP04;
HRCOMR05=HRCOMP05;
HRCOMR06=HRCOMP06;
HRCOMR07=HRCOMP07;

HRCOMR01=round(HRCOMR01,1);
HRCOMR02=round(HRCOMR02,1);
HRCOMR03=round(HRCOMR03,1);
HRCOMR04=round(HRCOMR04,1);
HRCOMR05=round(HRCOMR05,1);
HRCOMR06=round(HRCOMR06,1);
HRCOMR07=round(HRCOMR07,1);

endsas;
```

NUMBER OF WEEKS WORKED DURING 19XX/SINCE LAST INTERVIEW (OR BIRTH)/SINCE AGE 14

Variables Created: CV_WKSWK_YR.80 – CV_WKSWK_YR.98
 CV_WKSWK_DLI
 CV_WKSWK_EVER

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

This program creates three variables counting the number of weeks the respondent worked at any employee-type job during several different time periods.

1. CV_WKSWK_YR is created for each individual for each year of potential work activity (1980-98). Respondents not working in a given year are given a default value of zero (0) weeks worked. Otherwise, the variable the actual cumulative weeks worked on all jobs in that year.
2. CV_WKSWK_DLI is created for each individual since the last interview. In the first year, this variable measures the worked since birth.
3. CV_WKSWK_EVER is created for each individual to measure the number of weeks worked since the age of 14.

***** Overlay multiple jobs over JOB 1 work weeks.

This section is common to the 3 variables. *****;

```
do over job1wks; alljobs=job1wks; end;

do over alljobs;
if job2wks=1 then do; alljobs=job2wks; end;
if job2wks=-3 and alljobs=0 then do;
    alljobs=job2wks; end;
end;

do over alljobs;
if job3wks=1 then do; alljobs=job3wks; end;
if job3wks=-3 and alljobs=0 then do;
    alljobs=job3wks; end;
end;

do over alljobs;
if job4wks=1 then do; alljobs=job4wks; end;
if job4wks=-3 and alljobs=0 then do;
    alljobs=job4wks; end;
end;

do over alljobs;
if job5wks=1 then do; alljobs=job5wks; end;
if job5wks=-3 and alljobs=0 then do;
    alljobs=job5wks; end;
end;

do over alljobs;
if job6wks=1 then do; alljobs=job6wks; end;
if job6wks=-3 and alljobs=0 then do;
    alljobs=job6wks; end;
end;

do over alljobs;
if job7wks=1 then do; alljobs=job7wks; end;
if job7wks=-3 and alljobs=0 then do;
    alljobs=job7wks; end;
```

end;

***** 1. This section calculates cumulative weeks on all jobs for each year. *****

```
/* 1980 */
wks80=0;
do i=1 to 52;
    if alljobs=1 then do; wks80=wks80+1; end;
end;
do i=1 to 52;
    if alljobs=-3 then do; wks80=-3; end;
end;
```

*****At this point the program loops through the same code used above for 1980 for each year 1981–98, creating the variables wks81, wks82, wks83, and so on through wks98. These loops are deleted due to space considerations; users who need to see the entire code should contact NLS User Services. The week numbers for the “do i” statement for each year are as follows:

1981	53-104	1990	523-574
1982	105-156	1991	575-626
1983	157-209	1992	627-678
1984	210-261	1993	679-730
1985	262-313	1994	731-783
1986	314-365	1995	784-835
1987	366-417	1996	836-887
1988	418-470	1997	888-939
1989	471-522	1998	940-991 *****

```
do i=1 to 7;
/* start date invalid */
if starw<0 and starw>-4 then do;
    if stopw>1 then do; wks80=-3; end;
    if stopw>52 then do; wks81=-3; end;
    if stopw>104 then do; wks82=-3; end;
    if stopw>156 then do; wks83=-3; end;
    if stopw>209 then do; wks84=-3; end;
```

```

if stopw>261 then do; wks85=-3; end;
if stopw>313 then do; wks86=-3; end;
if stopw>365 then do; wks87=-3; end;
if stopw>417 then do; wks88=-3; end;
if stopw>470 then do; wks89=-3; end;
if stopw>522 then do; wks90=-3; end;
if stopw>574 then do; wks91=-3; end;
if stopw>626 then do; wks92=-3; end;
if stopw>678 then do; wks93=-3; end;
if stopw>730 then do; wks94=-3; end;
if stopw>783 then do; wks95=-3; end;
if stopw>834 then do; wks96=-3; end;
if stopw>887 then do; wks97=-3; end;
if stopw>939 then do; wks98=-3; end;
end;

/* stop date invalid */
if stopw<0 and stopw>-4 then do;
if starw<53 then do; wks80=-3; end;
if starw<105 then do; wks81=-3; end;
if starw<157 then do; wks82=-3; end;
if starw<210 then do; wks83=-3; end;
if starw<262 then do; wks84=-3; end;
if starw<314 then do; wks85=-3; end;
if starw<366 then do; wks86=-3; end;
if starw<418 then do; wks87=-3; end;
if starw<471 then do; wks88=-3; end;
if starw<523 then do; wks89=-3; end;
if starw<575 then do; wks90=-3; end;
if starw<627 then do; wks91=-3; end;
if starw<679 then do; wks92=-3; end;
if starw<731 then do; wks93=-3; end;
if starw<784 then do; wks94=-3; end;
if starw<836 then do; wks95=-3; end;
if starw<888 then do; wks96=-3; end;
if starw<940 then do; wks97=-3; end;
if starw<991 then do; wks98=-3; end;
end;
end;

*****2. This section calculates cumulative weeks on all jobs since the last interview. *****
allwks=0;

if birthwk>0 then do;
do i=birthwk to 992;
if alljobs=1 then do; allwks=allwks+1; end;
end;
do i=birthwk to 992;
if alljobs=-3 then do; allwks=-3; end;
end;
if starw1<0 and starw1>-4 then do; allwks=-3; end;
if starw2<0 and starw2>-4 then do; allwks=-3; end;
if starw3<0 and starw3>-4 then do; allwks=-3; end;
if starw4<0 and starw4>-4 then do; allwks=-3; end;
if starw5<0 and starw5>-4 then do; allwks=-3; end;

if starw6<0 and starw6>-4 then do; allwks=-3; end;
if starw7<0 and starw7>-4 then do; allwks=-3; end;
if stopw1<0 and stopw1>-4 then do; allwks=-3; end;
if stopw2<0 and stopw2>-4 then do; allwks=-3; end;
if stopw3<0 and stopw3>-4 then do; allwks=-3; end;
if stopw4<0 and stopw4>-4 then do; allwks=-3; end;
if stopw5<0 and stopw5>-4 then do; allwks=-3; end;
if stopw6<0 and stopw6>-4 then do; allwks=-3; end;
if stopw7<0 and stopw7>-4 then do; allwks=-3; end;
end;

/* [END CV_WKSWK_DLI] */

*****3. This section calculates cumulative weeks on all jobs since age 14. *****
wks14=0;

if age14wk>0 then do;
do i=age14wk to 992;
if alljobs=1 then do; wks14=wks14+1; end;
if alljobs=-3 then do; wks14=-3; end;
end;
end;

do i=1 to 7;
if age14wk<53 then do;
if starw<0 and starw>-4 and stopw>1 then do;
wks14=-3; end;
if stopw<0 and stopw>-4 and starw<53 then do;
wks14=-3; end;
end;
if age14wk<105 then do;
if starw<0 and starw>-4 and stopw>52 then do;
wks14=-3; end;
if stopw<0 and stopw>-4 and starw<105 then do;
wks14=-3; end;
end;
if age14wk<157 then do;
if starw<0 and starw>-4 and stopw>104 then do;
wks14=-3; end;
if stopw<0 and stopw>-4 and starw<157 then do;
wks14=-3; end;
end;
if age14wk<210 then do;
if starw<0 and starw>-4 and stopw>156 then do;
wks14=-3; end;
if stopw<0 and stopw>-4 and starw<210 then do;
wks14=-3; end;
end;
if age14wk<262 then do;
if starw<0 and starw>-4 and stopw>209 then do;
wks14=-3; end;
if stopw<0 and stopw>-4 and starw<262 then do;
wks14=-3; end;
end;
if age14wk<314 then do;

```

```

if starw<0 and starw>-4 and stopw>261 then do;
  wks14=-3; end;
if stopw<0 and stopw>-4 and starw<314 then do;
  wks14=-3; end;
end;
if age14wk<366 then do;
  if starw<0 and starw>-4 and stopw>313 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<366 then do;
    wks14=-3; end;
  end;
if age14wk<418 then do;
  if starw<0 and starw>-4 and stopw>365 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<418 then do;
    wks14=-3; end;
  end;
if age14wk<471 then do;
  if starw<0 and starw>-4 and stopw>417 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<471 then do;
    wks14=-3; end;
  end;
if age14wk<523 then do;
  if starw<0 and starw>-4 and stopw>470 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<523 then do;
    wks14=-3; end;
  end;
if age14wk<575 then do;
  if starw<0 and starw>-4 and stopw>522 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<575 then do;
    wks14=-3; end;
  end;
if age14wk<627 then do;
  if starw<0 and starw>-4 and stopw>574 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<627 then do;
    wks14=-3; end;
  end;
if age14wk<679 then do;
  if starw<0 and starw>-4 and stopw>626 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<679 then do;
    wks14=-3; end;
  end;
if age14wk<731 then do;
  if starw<0 and starw>-4 and stopw>678 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<731 then do;
    wks14=-3; end;
  end;
if age14wk<784 then do;
  if starw<0 and starw>-4 and stopw>730 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<784 then do;
    wks14=-3; end;
  end;
if age14wk<836 then do;
  if starw<0 and starw>-4 and stopw>783 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<836 then do;
    wks14=-3; end;
  end;
if age14wk<888 then do;
  if starw<0 and starw>-4 and stopw>835 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<888 then do;
    wks14=-3; end;
  end;
if age14wk<940 then do;
  if starw<0 and starw>-4 and stopw>887 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<940 then do;
    wks14=-3; end;
  end;
if age14wk<991 then do;
  if starw<0 and starw>-4 and stopw>939 then do;
    wks14=-3; end;
  if stopw<0 and stopw>-4 and starw<991 then do;
    wks14=-3; end;
  end;
end;
***** Include valid skips for all variables *****;

if e200=0 then do;
  wks80=-4; wks81=-4; wks82=-4; wks83=-4; wks84=-4;
  wks85=-4; wks86=-4; wks87=-4; wks88=-4;
  wks89=-4; wks90=-4; wks91=-4; wks92=-4;
  wks93=-4; wks94=-4; wks95=-4; wks96=-4;
  wks97=-4; wks98=-4; allwks=-4; wks14=-4; end;

if e200=-3 then do;
  wks80=-3; wks81=-3; wks82=-3; wks83=-3; wks84=-3;
  wks85=-3; wks86=-3; wks87=-3; wks88=-3;
  wks89=-3; wks90=-3; wks91=-3; wks92=-3;
  wks93=-3; wks94=-3; wks95=-3; wks96=-3;
  wks97=-3; wks98=-3; allwks=-3; wks14=-3; end;

*if start year is missing;
if pubid=411 or pubid=631 then do;
  wks80=-3; wks81=-3; wks82=-3; wks83=-3; wks84=-3;
  wks85=-3; wks86=-3; wks87=-3; wks88=-3;
  wks89=-3; wks90=-3; wks91=-3; wks92=-3;
  wks93=-3; wks94=-3; wks95=-3; wks96=-3;
  wks97=-3; wks98=-3; allwks=-3; wks14=-3; end;

endsas;

```

WEEKS WORKED AT EMPLOYEE JOB #X DURING 19XX

Variables Created: CV_WKSWK_JOB_YR.01.80 – CV_WKSWK_JOB_YR.01.98
CV_WKSWK_JOB_YR.02.80 – CV_WKSWK_JOB_YR.02.98 etc. through job #7

Programs Used

This program uses **emp_begin.sas** as input (see the first page of this appendix for details).

This program creates variables for each of the respondent's jobs counting the number of weeks worked in each calendar year. A variable is created for each potential job even if the respondent has worked no jobs in a given year with the default value set to zero (0). The most jobs held by any respondent in round 1 was seven, so variables are created for seven jobs for each respondent.

```
***** Calculate cumulative weeks on
individual jobs for each year *****;
/* 1980 */
wks801=0; wks802=0; wks803=0; wks804=0;
wks805=0; wks806=0; wks807=0;

do i=1 to 52;
  if job1wks=1 then do; wks801=wks801+1; end;
  if job2wks=1 then do; wks802=wks802+1; end;
  if job3wks=1 then do; wks803=wks803+1; end;
  if job4wks=1 then do; wks804=wks804+1; end;
  if job5wks=1 then do; wks805=wks805+1; end;
  if job6wks=1 then do; wks806=wks806+1; end;
  if job7wks=1 then do; wks807=wks807+1; end;
end;

do i=1 to 52;
  if job1wks=-3 then do; wks801=-3; end;
  if job2wks=-3 then do; wks802=-3; end;
  if job3wks=-3 then do; wks803=-3; end;
  if job4wks=-3 then do; wks804=-3; end;
  if job5wks=-3 then do; wks805=-3; end;
  if job6wks=-3 then do; wks806=-3; end;
  if job7wks=-3 then do; wks807=-3; end;
end;

***** At this point the program loops through the
same code used above for 1980 for each year 1981–98,
creating the variables wks811 through wks817 for
1981, wks821 through wks827 for 1982–82, and so on
up to wks981 through wks987 for 1998. These loops
are deleted due to space considerations; users who need
to see the entire code should contact NLS User
Services. The week numbers for the “do i” statement
for each year are as follows:
  1981  53-104      1990  523-574
  1982  105-156     1991  575-626
  1983  157-209     1992  627-678
  1984  210-261     1993  679-730
  1985  262-313     1994  731-783
  1986  314-365     1995  784-835
  1987  366-417     1996  836-887
  1988  418-470     1997  888-939
```

1989 471-522	1998 940-991 *****/
-----------------	------------------------

```
*** Insert valid skips;
if e200=0 then do;
  wks801=-4; wks802=-4; wks803=-4; wks804=-4;
  wks805=-4; wks806=-4; wks807=-4;
*** The seven weeks variables for each intervening year
  are similarly set to -4 ***
  wks981=-4; wks982=-4; wks983=-4; wks984=-4;
  wks985=-4; wks986=-4; wks987=-4;
end;

if e200=-3 then do;
  wks801=-3; wks802=-3; wks803=-3; wks804=-3;
  wks805=-3; wks806=-3; wks807=-3;
*** The seven weeks variables for each intervening year
  are similarly set to -3 ***
  wks981=-3; wks982=-3; wks983=-3; wks984=-3;
  wks985=-3; wks986=-3; wks987=-3;
end;

*if start year is missing;
if pubid=411 or pubid=631 then do;
  wks801=-3; wks811=-3; wks821=-3; wks831=-3;
  wks841=-3; wks851=-3; wks861=-3; wks871=-3;
  wks881=-3; wks891=-3; wks901=-3; wks911=-3;
  wks921=-3; wks931=-3; wks941=-3; wks951=-3;
  wks961=-3; wks971=-3; wks981=-3;
end;

if starw1<0 and starw1>-4 then do;

/*1980*/
  if stopw1>1 then do; wks801=-3; end;
  if stopw2>1 then do; wks802=-3; end;
  if stopw3>1 then do; wks803=-3; end;
  if stopw4>1 then do; wks804=-3; end;
  if stopw5>1 then do; wks805=-3; end;
  if stopw6>1 then do; wks806=-3; end;
  if stopw7>1 then do; wks807=-3; end;

*** The same seven commands applied to the 1980
  weeks variables are used for each subsequent year.
```

Appendix 2: Employment Variable Creation

However, only the first job (first weeks variable) for each year is shown here due to space considerations. **/

```

ifstopw1>52 then do; wks811=-3; end;
      /*** and so on through job 7 for 1981 ***/
if stopw1>104 then do; wks821=-3; end;
      /*** and so on through job 7 for 1982 ***/
if stopw1>156 then do; wks831=-3; end;
      /*** and so on through job 7 for 1983 ***/
if stopw1>209 then do; wks841=-3; end;
      /*** and so on through job 7 for 1984 ***/
if stopw1>261 then do; wks851=-3; end;
      /*** and so on through job 7 for 1985 ***/
if stopw1>313 then do; wks861=-3; end;
      /*** and so on through job 7 for 1986 ***/
if stopw1>365 then do; wks871=-3; end;
      /*** and so on through job 7 for 1987 ***/
if stopw1>417 then do; wks881=-3; end;
      /*** and so on through job 7 for 1988 ***/
if stopw1>470 then do; wks891=-3; end;
      /*** and so on through job 7 for 1989 ***/
if stopw1>522 then do; wks901=-3; end;
      /*** and so on through job 7 for 1990 ***/
if stopw1>574 then do; wks911=-3; end;
      /*** and so on through job 7 for 1991 ***/
if stopw1>626 then do; wks921=-3; end;
      /*** and so on through job 7 for 1992 ***/
if stopw1>678 then do; wks931=-3; end;
      /*** and so on through job 7 for 1993 ***/
if d stopw1>730 then do; wks941=-3; end;
      /*** and so on through job 7 for 1994 ***/
if stopw1>783 then do; wks951=-3; end;
      /*** and so on through job 7 for 1995 ***/
if stopw1>835 then do; wks961=-3; end;
      /*** and so on through job 7 for 1996 ***/
if stopw1>887 then do; wks971=-3; end;
      /*** and so on through job 7 for 1997 ***/
if stopw1>939 then do; wks981=-3; end;
      /*** and so on through job 7 for 1998 ***/

end;

/* stop date invalid */
if stopw1<0 and stopw1>-4 then do;

/*1980*/
  if starw1<53 then do; wks801=-3; end;
  if starw2<53 then do; wks802=-3; end;
  if starw3<53 then do; wks803=-3; end;

```

```

if starw4<53 then do; wks804=-3; end;
if starw5<53 then do; wks805=-3; end;
if starw6<53 then do; wks806=-3; end;
if starw7<53 then do; wks807=-3; end;

/*** Again, the same seven commands applied to the
1980 weeks variables are used for each subsequent year;
only the first job (first weeks variable) is shown here. */

if starw1<105 then do; wks811=-3; end;
      /*** and so on through job 7 for 1981 ***/
if starw1<157 then do; wks821=-3; end;
      /*** and so on through job 7 for 1982 ***/
if starw1<210 then do; wks831=-3; end;
      /*** and so on through job 7 for 1983 ***/
if starw1<262 then do; wks841=-3; end;
      /*** and so on through job 7 for 1984 ***/
if starw1<314 then do; wks851=-3; end;
      /*** and so on through job 7 for 1985 ***/
if starw1<366 then do; wks861=-3; end;
      /*** and so on through job 7 for 1986 ***/
if starw1<418 then do; wks871=-3; end;
      /*** and so on through job 7 for 1987 ***/
if starw1<471 then do; wks881=-3; end;
      /*** and so on through job 7 for 1988 ***/
if starw1<523 then do; wks891=-3; end;
      /*** and so on through job 7 for 1989 ***/
if starw1<575 then do; wks901=-3; end;
      /*** and so on through job 7 for 1990 ***/
if starw1<627 then do; wks911=-3; end;
      /*** and so on through job 7 for 1991 ***/
if starw1<679 then do; wks921=-3; end;
      /*** and so on through job 7 for 1992 ***/
if starw1<731 then do; wks931=-3; end;
      /*** and so on through job 7 for 1993 ***/
if starw1<784 then do; wks941=-3; end;
      /*** and so on through job 7 for 1994 ***/
if starw1<836 then do; wks951=-3; end;
      /*** and so on through job 7 for 1995 ***/
if starw1<888 then do; wks961=-3; end;
      /*** and so on through job 7 for 1996 ***/
if starw1<940 then do; wks971=-3; end;
      /*** and so on through job 7 for 1997 ***/
if starw1<991 then do; wks981=-3; end;
      /*** and so on through job 7 for 1998 ***/

end;

endsas;

```

TOTAL TENURE AT JOB #X AS OF THE SURVEY DATE

Variables Created: CV_WKSWK_JOB_DLI.01 – CV_WKSWK_JOB_DLI.07

Programs Used

This program uses **emp_begin.sas** as input (see the first page of this appendix for details).

This program creates a variable for each job calculating the total length of job tenure in weeks excluding within-job gaps. A variable is created for each potential job even if the respondent has no data for that job, with the default value set to zero (0). The most jobs held by any respondent in round 1 was seven, so variables are created for seven jobs for each respondent.

```
***** Calculate cumulative weeks on individual jobs for all years *****;

tenure1=0; tenure2=0; tenure3=0; tenure4=0;
tenure5=0; tenure6=0; tenure7=0;

do i=1 to 992;
  if job1wks=1 then do; tenure1=tenure1+1; end;
  if job2wks=1 then do; tenure2=tenure2+1; end;
  if job3wks=1 then do; tenure3=tenure3+1; end;
  if job4wks=1 then do; tenure4=tenure4+1; end;
  if job5wks=1 then do; tenure5=tenure5+1; end;
  if job6wks=1 then do; tenure6=tenure6+1; end;
  if job7wks=1 then do; tenure7=tenure7+1; end;
end;

flag=0;

do i=1 to 992;
  if job1wks=-3 then do; tenure1=-3; flag=1; end;
  if job2wks=-3 then do; tenure2=-3; flag=2; end;
  if job3wks=-3 then do; tenure3=-3; flag=3; end;
  if job4wks=-3 then do; tenure4=-3; flag=4; end;
  if job5wks=-3 then do; tenure5=-3; flag=5; end;
  if job6wks=-3 then do; tenure6=-3; flag=6; end;
  if job7wks=-3 then do; tenure7=-3; flag=7; end;
end;

do i=1 to 992;
  if starw1<0 and starw1>-4 then do; tenure1=-3; end;
  if starw2<0 and starw2>-4 then do; tenure2=-3; end;
  if starw3<0 and starw3>-4 then do; tenure3=-3; end;

```

```

if starw4<0 and starw4>-4 then do; tenure4=-3; end;
if starw5<0 and starw5>-4 then do; tenure5=-3; end;
if starw6<0 and starw6>-4 then do; tenure6=-3; end;
if starw7<0 and starw7>-4 then do; tenure7=-3; end;

if stopw1<0 and stopw1>-4 then do; tenure1=-3; end;
if stopw2<0 and stopw2>-4 then do; tenure2=-3; end;
if stopw3<0 and stopw3>-4 then do; tenure3=-3; end;
if stopw4<0 and stopw4>-4 then do; tenure4=-3; end;
if stopw5<0 and stopw5>-4 then do; tenure5=-3; end;
if stopw6<0 and stopw6>-4 then do; tenure6=-3; end;
if stopw7<0 and stopw7>-4 then do; tenure7=-3; end;
end;

if e200=0 then do;
  tenure1=-4; tenure2=-4; tenure3=-4; tenure4=-4;
  tenure5=-4; tenure6=-4; tenure7=-4;
end;

if e200=-3 then do;
  tenure1=-3; tenure2=-3; tenure3=-3; tenure4=-3;
  tenure5=-3; tenure6=-3; tenure7=-3;
end;

*if start year is missing;
if pubid=411 or pubid=631 then do; tenure1=-3; end;

endsas;

```

TOTAL HOURS WORKED IN 19XX/SINCE AGE 14

Variables Created: CV_HOURS_WK_YR.80 – CV_HOURS_WK_YR.98
CV_HOURS_WK_EVER

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

This program calculates the number of hours worked by the respondent at all employee-type jobs in each calendar year and since turning 14. A variable is created for each respondent even if the respondent has worked no jobs in a given year with the default value set to zero (0). Note that when both "starting hours" and "current hours" are reported, the latter are used to construct these measures

```
*****Organize hours for each job. This
section is common to both programs. *****

/* shrs161 = starting hours age<16
   hrwk161 = ending hours age<16
   shrs1 = starting hours age 16+
   hrwk1 = ending hours age 16+ */

array s16hrs (k) shrs161-shrs167;
array hr16wk (k) hrwk161-hrwk167;
array shrs (k) shrs1-shrs7;
array hrwk (k) hrwk1-hrwk7;

array hour16s (k) hours161-hours167;
array hours (k) hours1-hours7;
array hrck16 (k) hrck161-hrck167;
array hrck (k) hrck1-hrck7;

array starw (i) starw1-starw7;
array stopw (i) stopw1-stopw7;
* Set default hours to zero (0);
do over hours;
  hour16s=0;
  hours=0;
  hrck16=0;
  hrck=0;
end;

* Take starting hours if reported (eliminates -4's);
do over s16hrs;
  if s16hrs>0 then do; hour16s=s16hrs; end;
end;

do over s16hrs;
  if s16hrs<0 and s16hrs ne -4 and s16hrs ne . then do;
    hrck16=s16hrs; end;
end;

* Write over if end hours reported;
do over hr16wk;
  if hr16wk>0 then do; hour16s=hr16wk; end;
end;

do over hr16wk;

if hr16wk<0 and hr16wk ne -4 and hr16wk ne . then
  do; hrck16=hr16wk; end;
if hr16wk=0 or hr16wk>0 then do; hrck16=0; end;
end;

* Take starting hours if reported (eliminates -4's);
do over shrs;
  if shrs>0 then do; hours=shrs; end;
end;

do over shrs;
  if shrs<0 and shrs ne -4 and shrs ne . then do;
    hrck=shrs; end;
end;

* Write over if end hours reported;
do over hrwk;
  if hrwk>0 then do; hours=hrwk; end;
end;

do over hrwk;
  if hrwk<0 and hrwk ne -4 and hrwk ne . then do;
    hrck=hrwk; end;
  if hrwk=0 or hrwk>0 then do; hrck=0; end;
end;

***** 1. This section calculates
cumulative weeks on each job for each year. Ends at:
[END CV_HOURS_WK_YR] *****

/* 1980 */
wks16801=0; wks16802=0; wks16803=0;
wks16804=0; wks16805=0; wks16806=0;
wks16807=0;

do i=1 to 52;
  if job1wks=1 then do; wks16801=wks16801+1; end;
  if job2wks=1 then do; wks16802=wks16802+1; end;
  if job3wks=1 then do; wks16803=wks16803+1; end;
  if job4wks=1 then do; wks16804=wks16804+1; end;
  if job5wks=1 then do; wks16805=wks16805+1; end;
  if job6wks=1 then do; wks16806=wks16806+1; end;
  if job7wks=1 then do; wks16807=wks16807+1; end;
end;
```

```

do i=1 to 52;
  if job1wks ne -3 and job2wks ne -3 and job3wks ne -
    3 and job4wks ne -3 and job5wks ne -3 and
    job6wks ne -3 and job7wks ne -3 then do;
    ah16801=hours161*wks16801;
    ah16802=hours162*wks16802;
    ah16803=hours163*wks16803;
    ah16804=hours164*wks16804;
    ah16805=hours165*wks16805;
    ah16806=hours166*wks16806;
    ah16807=hours167*wks16807;
  end;
  if job1wks=-3 or job2wks=-3 or job3wks=-3 or
    job4wks=-3 or job5wks=-3 or job6wks=-3 or
    job7wks=-3 then do;
    ah16801=-3; ah16802=-3; ah16803=-3;
    ah16804=-3; ah16805=-3; ah16806=-3;
    ah16807=-3;
  goto exit1;
  end;
end;
exit1:

if ah16801 ne -3 and ah16802 ne -3 and ah16803 ne -
  3 and ah16804 ne -3 and ah16805 ne -3 and
  ah16806 ne -3 and ah16807 ne -3 then do;
  tothrs80=ah16801+ah16802+ah16803+
    ah16804+ah16805+ah16806+ah16807;
end;
if ah16801=-3 or ah16802=-3 or ah16803=-3 or
  ah16804=-3 or ah16805=-3 or ah16806=-3 or
  ah16807=-3 then do;
  tothrs80=-3;
end;

if wks16801>0 and hrck161<0 then tothrs80=hrck161;
if wks16802>0 and hrck162<0 then tothrs80=hrck162;
if wks16803>0 and hrck163<0 then tothrs80=hrck163;
if wks16804>0 and hrck164<0 then tothrs80=hrck164;
if wks16805>0 and hrck165<0 then tothrs80=hrck165;
if wks16806>0 and hrck166<0 then tothrs80=hrck166;
if wks16807>0 and hrck167<0 then tothrs80=hrck167;

***** At this point the program loops through the
same code used above for 1980 for each year 1981–85,
creating the variables wks16811 through 16817 for
1981, wks16821 through wks16827 for 1982, and so
on through 1985 (at which point the pattern changes).
These loops are deleted due to space considerations;
users who need to see the entire code should contact
NLS User Services. The week numbers used in the “do
i” statement for 1981–85 are as follows:
  1981  53-104
  1982  105-156
  1983  157-209
  1984  210-261
  1985  262-313*****

```

***** Beginning in 1986 the creation pattern changes due to different questions asked of those age 16 and older. The complete code for 1986 is shown here; the similar code for 1987–98 is deleted due to space considerations. The week numbers used in the “do i” statements for 1987–98 are as follows:

1987	366-417	1993	679-730
1988	418-470	1994	731-783
1989	471-522	1995	784-835
1990	523-574	1996	836-887
1991	575-626	1997	888-939
1992	627-678	1998	940-991

```

/* 1986 */
wks861=0; wks862=0; wks863=0; wks864=0;
  wks865=0; wks866=0; wks867=0;
wks16861=0; wks16862=0; wks16863=0;
  wks16864=0; wks16865=0; wks16866=0;
  wks16867=0;

```

```

do i=314 to 365;
  if job1wks=1 then do; wks861=wks861+1; end;
  if job2wks=1 then do; wks862=wks862+1; end;
  if job3wks=1 then do; wks863=wks863+1; end;
  if job4wks=1 then do; wks864=wks864+1; end;
  if job5wks=1 then do; wks865=wks865+1; end;
  if job6wks=1 then do; wks866=wks866+1; end;
  if job7wks=1 then do; wks867=wks867+1; end;
end;

```

```

do i=314 to 365;
  if job1wks=1 then do; wks16861=wks16861+1; end;
  if job2wks=1 then do; wks16862=wks16862+1; end;
  if job3wks=1 then do; wks16863=wks16863+1; end;
  if job4wks=1 then do; wks16864=wks16864+1; end;
  if job5wks=1 then do; wks16865=wks16865+1; end;
  if job6wks=1 then do; wks16866=wks16866+1; end;
  if job7wks=1 then do; wks16867=wks16867+1; end;
end;

```

```

do i=314 to 365;
  if job1wks ne -3 and job2wks ne -3 and job3wks ne -
    3 and job4wks ne -3 and job5wks ne -3 and
    job6wks ne -3 and job7wks ne -3 then do;
    ah16861=hours161*wks16861;
    ah16862=hours162*wks16862;
    ah16863=hours163*wks16863;
    ah16864=hours164*wks16864;
    ah16865=hours165*wks16865;
    ah16866=hours166*wks16866;
    ah16867=hours167*wks16867;
    ah861=hours1*wks861;
    ah862=hours2*wks862;
    ah863=hours3*wks863;
    ah864=hours4*wks864;
    ah865=hours5*wks865;

```

Appendix 2: Employment Variable Creation

```

ah866=hours6*wks866;
ah867=hours7*wks867;
end;
if job1wks=-3 or job2wks=-3 or job3wks=-3 or
    job4wks=-3 or job5wks=-3 or job6wks=-3 or
    job7wks=-3 then do;
ah16861=-3; ah16862=-3; ah16863=-3;
    ah16864=-3; ah16865=-3; ah16866=-3;
    ah16867=-3;
ah861=-3; ah862=-3; ah863=-3; ah864=-3;
    ah865=-3; ah866=-3; ah867=-3;
    goto exit7;
end;
end; exit7;

if ah16861 ne -3 and ah16862 ne -3 and ah16863 ne -
    3 and ah16864 ne -3 and ah16865 ne -3 and
    ah16866 ne -3 and ah16867 ne -3 and
ah861 ne -3 and ah862 ne -3 and ah863 ne -3 and
    ah864 ne -3 and ah865 ne -3 and ah866 ne -3
    and ah867 ne -3 then do;
tohrs86=ah16861+ah16862+ah16863+ah16864+
    ah16865+ah16866+ah16867+ah861+
    ah862+ah863+ah864+ah865+ah866+ ah867;
end;
if ah16861=-3 or ah16862=-3 or ah16863=-3 or
    ah16864=-3 or ah16865=-3 or ah16866=-3 or
    ah16867=-3 or
ah861=-3 or ah862=-3 or ah863=-3 or ah864=-3 or
    ah865=-3 or ah866=-3 or ah867=-3 then do;
    tohrs86=-3;
end;

if wks16861>0 and hrck161<0 then tohrs86=hrck161;
if wks16862>0 and hrck162<0 then tohrs86=hrck162;
if wks16863>0 and hrck163<0 then tohrs86=hrck163;
if wks16864>0 and hrck164<0 then tohrs86=hrck164;
if wks16865>0 and hrck165<0 then tohrs86=hrck165;
if wks16866>0 and hrck166<0 then tohrs86=hrck166;
if wks16867>0 and hrck167<0 then tohrs86=hrck167;

if wks861>0 and hrck1<0 then tohrs86=hrck1;
if wks862>0 and hrck2<0 then tohrs86=hrck2;
if wks863>0 and hrck3<0 then tohrs86=hrck3;
if wks864>0 and hrck4<0 then tohrs86=hrck4;
if wks865>0 and hrck5<0 then tohrs86=hrck5;
if wks866>0 and hrck6<0 then tohrs86=hrck6;
if wks867>0 and hrck7<0 then tohrs86=hrck7;

/* 1987-98 deleted */

do i=1 to 7;
/* start date invalid */
if starw<0 and starw>-4 then do;
if stopw>1 then do; tohrs80=-3; end;
if stopw>52 then do; tohrs81=-3; end;
if stopw>104 then do; tohrs82=-3; end;
if stopw>156 then do; tohrs83=-3; end;
if stopw>209 then do; tohrs84=-3; end;
if stopw>261 then do; tohrs85=-3; end;
if stopw>313 then do; tohrs86=-3; end;
if stopw>365 then do; tohrs87=-3; end;
if stopw>417 then do; tohrs88=-3; end;
if stopw>470 then do; tohrs89=-3; end;
if stopw>522 then do; tohrs90=-3; end;
if stopw>574 then do; tohrs91=-3; end;
if stopw>626 then do; tohrs92=-3; end;
if stopw>678 then do; tohrs93=-3; end;
if stopw>730 then do; tohrs94=-3; end;
if stopw>783 then do; tohrs95=-3; end;
if stopw>835 then do; tohrs96=-3; end;
if stopw>887 then do; tohrs97=-3; end;
if stopw>939 then do; tohrs98=-3; end;
end;

/* stop date invalid */
if stopw<0 and stopw>-4 then do;
if starw<53 then do; tohrs80=-3; end;
if starw<105 then do; tohrs81=-3; end;
if starw<157 then do; tohrs82=-3; end;
if starw<210 then do; tohrs83=-3; end;
if starw<262 then do; tohrs84=-3; end;
if starw<314 then do; tohrs85=-3; end;
if starw<366 then do; tohrs86=-3; end;
if starw<418 then do; tohrs87=-3; end;
if starw<471 then do; tohrs88=-3; end;
if starw<523 then do; tohrs89=-3; end;
if starw<575 then do; tohrs90=-3; end;
if starw<627 then do; tohrs91=-3; end;
if starw<679 then do; tohrs92=-3; end;
if starw<731 then do; tohrs93=-3; end;
if starw<784 then do; tohrs94=-3; end;
if starw<836 then do; tohrs95=-3; end;
if starw<888 then do; tohrs96=-3; end;
if starw<940 then do; tohrs97=-3; end;
if starw<991 then do; tohrs98=-3; end;
end;
end;

/* [END CV_HOURS_WK_YR] */

***** 2. This section calculates
cumulative weeks on all jobs since age 14. Ends at:
[END CV_HOURS_WK_EVER] *****;

wks1=0; wks2=0; wks3=0; wks4=0; wks5=0; wks6=0;
wks7=0;

if 0<age14wk<992 then do;
do i=age14wk to 992;
    if job1wks=1 then do; wks1=wks1+1; end;
    if job2wks=1 then do; wks2=wks2+1; end;
    if job3wks=1 then do; wks3=wks3+1; end;
    if job4wks=1 then do; wks4=wks4+1; end;

```

Appendix 2: Employment Variable Creation

```

if job5wks=1 then do; wks5=wks5+1; end;
if job6wks=1 then do; wks6=wks6+1; end;
if job7wks=1 then do; wks7=wks7+1; end;
end;
end;

if 0<age14wk<992 then do;
do i=age14wk to 991;
if job1wks ne -3 and job2wks ne -3 and job3wks ne
-3 and job4wks ne -3 and job5wks ne -3 and
job6wks ne -3 and job7wks ne -3 then do;
thrs161=hours161*wks1;
thrs162=hours162*wks2;
thrs163=hours163*wks3;
thrs164=hours164*wks4;
thrs165=hours165*wks5;
thrs166=hours166*wks6;
thrs167=hours167*wks7;
thrs1=hours1*wks1;
thrs2=hours2*wks2;
thrs3=hours3*wks3;
thrs4=hours4*wks4;
thrs5=hours5*wks5;
thrs6=hours6*wks6;
thrs7=hours7*wks7;
end;
if job1wks=-3 or job2wks=-3 or job3wks=-3 or
job4wks=-3 or job5wks=-3 or job6wks=-3 or
job7wks=-3 then do;
thrs1=-3;
tohrs14=-3;
goto exit1;
end;
end;
end; exit1:

if thrs161 ne -3 and thrs162 ne -3 and thrs163 ne -3
and thrs164 ne -3 and thrs165 ne -3 and
thrs166 ne -3 and thrs167 ne -3 and
thrs1 ne -3 and thrs2 ne -3 and thrs3 ne -3 and thrs4
ne -3 and thrs5 ne -3 and thrs6 ne -3 and
thrs7 ne -3 then do;
othrs14=thrs161+thrs162+thrs163+thrs164+thrs165+
thrs166+thrs167+thrs1+thrs2+thrs3+thrs4+
thrs5+thrs6+thrs7;
end;
if thrs161=-3 or thrs162=-3 or thrs163=-3 or thrs164=-3
or thrs165=-3 or thrs166=-3 or thrs167=-3 or
thrs1=-3 or thrs2=-3 or thrs3=-3 or thrs4=-3 or
thrs5=-3 or thrs6=-3 or thrs7=-3 then do;
tohrs14=-3;
end;

if wks1>0 and hrck161<0 and tohrs14>0 then
tohrs14=hrck161;
if wks2>0 and hrck162<0 and tohrs14>0 then
tohrs14=hrck162;
if wks3>0 and hrck163<0 and tohrs14>0 then
tohrs14=hrck163;
if wks4>0 and hrck164<0 and tohrs14>0 then
tohrs14=hrck164;
if wks5>0 and hrck165<0 and tohrs14>0 then
tohrs14=hrck165;
if wks6>0 and hrck166<0 and tohrs14>0 then
tohrs14=hrck166;
if wks7>0 and hrck167<0 and tohrs14>0 then
tohrs14=hrck167;

if wks1>0 and hrck1<0 and tohrs14>0 then
tohrs14=hrck1;
if wks2>0 and hrck2<0 and tohrs14>0 then
tohrs14=hrck2;
if wks3>0 and hrck3<0 and tohrs14>0 then
tohrs14=hrck3;
if wks4>0 and hrck4<0 and tohrs14>0 then
tohrs14=hrck4;
if wks5>0 and hrck5<0 and tohrs14>0 then
tohrs14=hrck5;
if wks6>0 and hrck6<0 and tohrs14>0 then
tohrs14=hrck6;
if wks7>0 and hrck7<0 and tohrs14>0 then
tohrs14=hrck7;

do i=1 to 7;
if age14wk<53 then do;
if starw<0 and starw>-4 and stopw>1 then do;
tohrs14=-3; end;
if stopw<0 and stopw>-4 and starw<53 then do;
tohrs14=-3; end;
end;
if age14wk<105 then do;
if starw<0 and starw>-4 and stopw>52 then do;
tohrs14=-3; end;
if stopw<0 and stopw>-4 and starw<105 then do;
tohrs14=-3; end;
end;
if age14wk<157 then do;
if starw<0 and starw>-4 and stopw>104 then do;
tohrs14=-3; end;
if stopw<0 and stopw>-4 and starw<157 then do;
tohrs14=-3; end;
end;
if age14wk<210 then do;
if starw<0 and starw>-4 and stopw>156 then do;
tohrs14=-3; end;
if stopw<0 and stopw>-4 and starw<210 then do;
tohrs14=-3; end;
end;
if age14wk<262 then do;
if starw<0 and starw>-4 and stopw>209 then do;
tohrs14=-3; end;
if stopw<0 and stopw>-4 and starw<262 then do;
tohrs14=-3; end;
end;

```

```

if age14wk<314 then do;
  if starw<0 and starw>-4 and stopw>261 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<314 then do;
    tothrs14=-3; end;
end;
if age14wk<366 then do;
  if starw<0 and starw>-4 and stopw>313 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<366 then do;
    tothrs14=-3; end;
end;
if age14wk<418 then do;
  if starw<0 and starw>-4 and stopw>365 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<418 then do;
    tothrs14=-3; end;
end;
if age14wk<471 then do;
  if starw<0 and starw>-4 and stopw>417 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<471 then do;
    tothrs14=-3; end;
end;
if age14wk<523 then do;
  if starw<0 and starw>-4 and stopw>470 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<523 then do;
    tothrs14=-3; end;
end;
if age14wk<575 then do;
  if starw<0 and starw>-4 and stopw>522 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<575 then do;
    tothrs14=-3; end;
end;
if age14wk<627 then do;
  if starw<0 and starw>-4 and stopw>574 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<627 then do;
    tothrs14=-3; end;
end;
if age14wk<679 then do;
  if starw<0 and starw>-4 and stopw>626 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<679 then do;
    tothrs14=-3; end;
end;
if age14wk<731 then do;
  if starw<0 and starw>-4 and stopw>678 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<731 then do;
    tothrs14=-3; end;
end;
if age14wk<784 then do;
  if starw<0 and starw>-4 and stopw>730 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<784 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<836 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<888 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<940 then do;
    tothrs14=-3; end;
  if stopw<0 and stopw>-4 and starw<991 then do;
    tothrs14=-3; end;
end;
/* [END CV_HOURS_WK_EVER] */

if e200=0 then do;
  tothrs80=-4; tothrs81=-4; tothrs82=-4; tothrs83=-4;
  tothrs84=-4; tothrs85=-4; tothrs86=-4; tothrs87=-4;
  tothrs88=-4; tothrs89=-4; tothrs90=-4; tothrs91=-4;
  tothrs92=-4; tothrs93=-4; tothrs94=-4; tothrs95=-4;
  tothrs96=-4; tothrs97=-4; tothrs98=-4; tothrs14=-4;
end;
if e200=-3 then do;
  tothrs80=-3; tothrs81=-3; tothrs82=-3; tothrs83=-3;
  tothrs84=-3; tothrs85=-3; tothrs86=-3; tothrs87=-3;
  tothrs88=-3; tothrs89=-3; tothrs90=-3; tothrs91=-3;
  tothrs92=-3; tothrs93=-3; tothrs94=-3; tothrs95=-3;
  tothrs96=-3; tothrs97=-3; tothrs98=-3; tothrs14=-3;
end;
*if start year is missing;
if pubid=411 or pubid=631 then do;
  tothrs80=-3; tothrs81=-3; tothrs82=-3; tothrs83=-3;
  tothrs84=-3; tothrs85=-3; tothrs86=-3; tothrs87=-3;
  tothrs88=-3; tothrs89=-3; tothrs90=-3; tothrs91=-3;
  tothrs92=-3; tothrs93=-3; tothrs94=-3; tothrs95=-3;
  tothrs96=-3; tothrs97=-3; tothrs98=-3; tothrs14=-3;
end;
endsas;

```

NUMBER OF JOBS HELD DURING 19XX

Variables Created: CV_TTL_JOBS_YR.80 – CV_TTL_JOBS_YR.98

Programs Used

This program uses **emp_begin.sas** as input (see the first page of this appendix for details).

This program calculates the number of employee-type jobs the respondent held during each calendar year. This variable is created only for respondents who have worked at least one week since age 14.

```
***** Indicate if worked at least one
week on a job in a given year *****;
/* 1980 */
job801=0; job802=0; job803=0; job804=0; job805=0;
job806=0; job807=0;

do i=1 to 52;
if job1wks=-3 then do; job801=-3; end;
if job2wks=-3 then do; job802=-3; end;
if job3wks=-3 then do; job803=-3; end;
if job4wks=-3 then do; job804=-3; end;
if job5wks=-3 then do; job805=-3; end;
if job6wks=-3 then do; job806=-3; end;
if job7wks=-3 then do; job807=-3; end;
end;

do i=1 to 52;
if job1wks=1 then do; job801=1; end;
if job2wks=1 then do; job802=1; end;
if job3wks=1 then do; job803=1; end;
if job4wks=1 then do; job804=1; end;
if job5wks=1 then do; job805=1; end;
if job6wks=1 then do; job806=1; end;
if job7wks=1 then do; job807=1; end;
end;

if job801 ne -3 and job802 ne -3 and job803 ne -3 and
job804 ne -3 and job805 ne -3 and job806 ne -3 and
job807 ne -3 then do;
njobs80=sum(job801, job802, job803, job804,
job805, job806, job807);
end;

if job801=-3 or job802=-3 or job803=-3 or job804=-3
or job805=-3 or job806=-3 or job807=-3 then do;
njobs80=-3;
end;

*****At this point the program loops through the
same code used above for 1980 for each year 1981–98,
creating the variables job811 through job817 for 1981,
job821 through job827 for 1982, and so on up to
job981 through job987 for 1998. These loops are
deleted due to space considerations; users who need to
see the entire code should contact NLS User Services.
```

The week numbers for the “do i” statement for each year are as follows:

1981	53-104	1990	523-574
1982	105-156	1991	575-626
1983	157-209	1992	627-678
1984	210-261	1993	679-730
1985	262-313	1994	731-783
1986	314-365	1995	784-835
1987	366-417	1996	836-887
1988	418-470	1997	888-939
1989	471-522	1998	940-991 *****/

```
do i=1 to 7;
/* start date invalid */
if starw<0 and starw>-4 then do;
if stopw>1 then do; njobs80=njobs80+1; end;
if stopw>52 then do; njobs81=njobs81+1; end;
if stopw>104 then do; njobs82=njobs82+1; end;
if stopw>156 then do; njobs83=njobs83+1; end;
if stopw>209 then do; njobs84=njobs84+1; end;
if stopw>261 then do; njobs85=njobs85+1; end;
if stopw>313 then do; njobs86=njobs86+1; end;
if stopw>365 then do; njobs87=njobs87+1; end;
if stopw>417 then do; njobs88=njobs88+1; end;
if stopw>470 then do; njobs89=njobs89+1; end;
if stopw>522 then do; njobs90=njobs90+1; end;
if stopw>574 then do; njobs91=njobs91+1; end;
if stopw>626 then do; njobs92=njobs92+1; end;
if stopw>678 then do; njobs93=njobs93+1; end;
if stopw>730 then do; njobs94=njobs94+1; end;
if stopw>783 then do; njobs95=njobs95+1; end;
if stopw>835 then do; njobs96=njobs96+1; end;
if stopw>887 then do; njobs97=njobs97+1; end;
if stopw>939 and intwk>939 then do;
njobs98=njobs98+1; end;
end;
```

```
/*stop date invalid*/
if stopw<0 and stopw>-4 then do;
if starw<53 then do; njobs80=njobs80+1; end;
if starw<105 then do; njobs81=njobs81+1; end;
if starw<157 then do; njobs82=njobs82+1; end;
if starw<210 then do; njobs83=njobs83+1; end;
if starw<262 then do; njobs84=njobs84+1; end;
if starw<314 then do; njobs85=njobs85+1; end;
if starw<366 then do; njobs86=njobs86+1; end;
if starw<418 then do; njobs87=njobs87+1; end;
```

Appendix 2: Employment Variable Creation

```
if starw<471 then do; njobs88=njobs88+1; end;
if starw<523 then do; njobs89=njobs89+1; end;
if starw<575 then do; njobs90=njobs90+1; end;
if starw<627 then do; njobs91=njobs91+1; end;
if starw<679 then do; njobs92=njobs92+1 end;
if starw<731 then do; njobs93=njobs93+1; end;
if starw<784 then do; njobs94=njobs94+1; end;
if starw<836 then do; njobs95=njobs95+1; end;
if starw<888 then do; njobs96=njobs96+1; end;
if starw<940 then do; njobs97=njobs97+1; end;
if starw<991 and intwk>939 then do;
    njobs98=njobs98+1; end;
end;
end;

*** Include valid skips;

if e200=0 then do;
    njobs80=-4; njobs81=-4; njobs82=-4; njobs83=-4;
    njobs84=-4; njobs85=-4; njobs86=-4; njobs87=-4;
    njobs88=-4; njobs89=-4; njobs90=-4; njobs91=-4;
    njobs92=-4; njobs93=-4; njobs94=-4; njobs95=-4;
    njobs96=-4; njobs97=-4; njobs98=-4;
end;
endsas;
```

njobs92=-4; njobs93=-4; njobs94=-4; njobs95=-4;
njobs96=-4; njobs97=-4; njobs98=-4;
end;

if e200=-3 then do;
 njobs80=-3; njobs81=-3; njobs82=-3; njobs83=-3;
 njobs84=-3; njobs85=-3; njobs86=-3; njobs87=-3;
 njobs88=-3; njobs89=-3; njobs90=-3; njobs91=-3;
 njobs92=-3; njobs93=-3; njobs94=-3; njobs95=-3;
 njobs96=-3; njobs97=-3; njobs98=-3;
end;

*if start year is missing;
if pubid=411 or pubid=631 then do;
 njobs80=-3; njobs81=-3; njobs82=-3; njobs83=-3;
 njobs84=-3; njobs85=-3; njobs86=-3; njobs87=-3;
 njobs88=-3; njobs89=-3; njobs90=-3; njobs91=-3;
 njobs92=-3; njobs93=-3; njobs94=-3; njobs95=-3;
 njobs96=-3; njobs97=-3; njobs98=-3;
end;

TOTAL NUMBER OF JOBS HELD SINCE AGE 14

Variables Created: CV_TTL_JOBS_EVER

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

This program calculates the total number of employee-type jobs held by the respondent since age14. It is only created for respondents who have worked at least one week since age 14.

```
***** Indicate if worked at least one week on a given job since age 14 ****;
job1=0; job2=0; job3=0; job4=0; job5=0; job6=0; job7=0;

if age14wk>0 then do;
  do i=age14wk to 992;
    if job1wks=1 then do; job1=1; end;
    if job2wks=1 then do; job2=1; end;
    if job3wks=1 then do; job3=1; end;
    if job4wks=1 then do; job4=1; end;
    if job5wks=1 then do; job5=1; end;
    if job6wks=1 then do; job6=1; end;
    if job7wks=1 then do; job7=1; end;
  end;
end;

njobs14=sum(job1,job2,job3,job4,job5,job6,job7);

if starw1>-4 and starw1<0 and stopw1>0 and stopw1>age14wk then do; njobs14=njobs14+1; end;
if starw2>-4 and starw2<0 and stopw2>0 and stopw2>age14wk then do; njobs14=njobs14+1; end;
if starw3>-4 and starw3<0 and stopw3>0 and stopw3>age14wk then do; njobs14=njobs14+1; end;
if starw4>-4 and starw4<0 and stopw4>0 and stopw4>age14wk then do; njobs14=njobs14+1; end;
if starw5>-4 and starw5<0 and stopw5>0 and stopw5>age14wk then do; njobs14=njobs14+1; end;
if starw6>-4 and starw6<0 and stopw6>0 and stopw6>age14wk then do; njobs14=njobs14+1; end;
if starw7>-4 and starw7<0 and stopw7>0 and stopw7>age14wk then do; njobs14=njobs14+1; end;

if stopw1>-4 and stopw1<0 and starw1=>age14wk then do; njobs14=njobs14+1; end;
if stopw2>-4 and stopw2<0 and starw2=>age14wk then do; njobs14=njobs14+1; end;
if stopw3>-4 and stopw3<0 and starw3=>age14wk then do; njobs14=njobs14+1; end;
if stopw4>-4 and stopw4<0 and starw4=>age14wk then do; njobs14=njobs14+1; end;
if stopw5>-4 and stopw5<0 and starw5=>age14wk then do; njobs14=njobs14+1; end;
if stopw6>-4 and stopw6<0 and starw6=>age14wk then do; njobs14=njobs14+1; end;
if stopw7>-4 and stopw7<0 and starw7=>age14wk then do; njobs14=njobs14+1; end;

if stopw1>-4 and stopw1<0 and starw1<age14wk then do; njobs14=-3; end;
if stopw2>-4 and stopw2<0 and starw2<age14wk then do; njobs14=-3; end;
if stopw3>-4 and stopw3<0 and starw3<age14wk then do; njobs14=-3; end;
if stopw4>-4 and stopw4<0 and starw4<age14wk then do; njobs14=-3; end;
if stopw5>-4 and stopw5<0 and starw5<age14wk then do; njobs14=-3; end;
if stopw6>-4 and stopw6<0 and starw6<age14wk then do; njobs14=-3; end;
if stopw7>-4 and stopw7<0 and starw7<age14wk then do; njobs14=-3; end;

if e200=0 then do; njobs14=-4; end;
if e200=-3 then do; njobs14=-3; end;
/*if start year is missing */; if pubid=411 or pubid=631 then do; njobs14=-3; end;

endsas;
```

NLSY97 Appendix 3:
Family Background Variable Creation

HOUSEHOLD SIZE AS OF THE SURVEY DATE

Variables Created: CV_HH_SIZE
CV_HH_UNDER_6
CV_HH_UNDER_18

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
PUBID	PUBID	SE3001-16	SE-30.01-.16
SE1D, M, Y	SE-1_D,_M,_Y	SE3101-16	SE-31.01-.16
SE28D1-D16	SE-28.01_D-.16_D	SE3201-10	SE-32.01-.10
SE28M1-M16	SE-28.01_M-.16_M	SE3401-08	SE-34.01-.08
SE28Y1-Y16	SE-28.01_Y-.16_Y	SIDCODE	SIDCODE

This program creates several variables describing the composition of the respondent's household: the total number of residents, the number of residents under age 6, and the number of residents under age 18.

```
/* First, create a variable indicating the number of people in the household: */
array SE28D SE28D1 SE28D2 SE28D3 SE28D4 SE28D5 SE28D6 SE28D7 SE28D8 SE28D9 SE28D10
      SE28D11 SE28D12 SE28D13 SE28D14 SE28D15 SE28D16;
array SE28M SE28M1 SE28M2 SE28M3 SE28M4 SE28M5 SE28M6 SE28M7 SE28M8 SE28M9 SE28M10
      SE28M11 SE28M12 SE28M13 SE28M14 SE28M15 SE28M16;
array SE28Y SE28Y1 SE28Y2 SE28Y3 SE28Y4 SE28Y5 SE28Y6 SE28Y7 SE28Y8 SE28Y9 SE28Y10 SE28Y11
      SE28Y12 SE28Y13 SE28Y14 SE28Y15 SE28Y16;

if SE28D1 ne -4 or SE28M1 ne -4 or SE28Y1 ne -4 then dumhh1=1; else dumhh1=0;
if dumhh1=1 and SE28D2 ne -4 and SE28M2 ne -4 and SE28Y2 ne -4 then dumhh2=1; else dumhh2=0;
if dumhh2=1 and SE28D3 ne -4 and SE28M3 ne -4 and SE28Y3 ne -4 then dumhh3=1; else dumhh3=0;
if dumhh3=1 and SE28D4 ne -4 and SE28M4 ne -4 and SE28Y4 ne -4 then dumhh4=1; else dumhh4=0;
if dumhh4=1 and SE28D5 ne -4 and SE28M5 ne -4 and SE28Y5 ne -4 then dumhh5=1; else dumhh5=0;
if dumhh5=1 and SE28D6 ne -4 and SE28M6 ne -4 and SE28Y6 ne -4 then dumhh6=1; else dumhh6=0;
if dumhh6=1 and SE28D7 ne -4 and SE28M7 ne -4 and SE28Y7 ne -4 then dumhh7=1; else dumhh7=0;
if dumhh7=1 and SE28D8 ne -4 and SE28M8 ne -4 and SE28Y8 ne -4 then dumhh8=1; else dumhh8=0;
if dumhh8=1 and SE28D9 ne -4 and SE28M9 ne -4 and SE28Y9 ne -4 then dumhh9=1; else dumhh9=0;
if dumhh9=1 and SE28D10 ne -4 and SE28M10 ne -4 and SE28Y10 ne -4 then dumhh10=1; else dumhh10=0;
if dumhh10=1 and SE28D11 ne -4 and SE28M11 ne -4 and SE28Y11 ne -4 then dumhh11=1; else dumhh11=0;
if dumhh11=1 and SE28D12 ne -4 and SE28M12 ne -4 and SE28Y12 ne -4 then dumhh12=1; else dumhh12=0;
if dumhh12=1 and SE28D13 ne -4 and SE28M13 ne -4 and SE28Y13 ne -4 then dumhh13=1; else dumhh13=0;
if dumhh13=1 and SE28D14 ne -4 and SE28M14 ne -4 and SE28Y14 ne -4 then dumhh14=1; else dumhh14=0;
if dumhh14=1 and SE28D15 ne -4 and SE28M15 ne -4 and SE28Y15 ne -4 then dumhh15=1; else dumhh15=0;
if dumhh15=1 and SE28D16 ne -4 and SE28M16 ne -4 and SE28Y16 ne -4 then dumhh16=1; else dumhh16=0;

hhmember=dumhh1+dumhh2+dumhh3+dumhh4+dumhh5+dumhh6+dumhh7+dumhh8+dumhh9+dumhh10+du
mhh11+dumhh12+dumhh13+dumhh14+dumhh15+dumhh16;

/*Second, count the number of members under or above a given age. First compare their reported birth date with
the interview date and then count the number of members in the given age range. */

array age age01 age02 age03 age04 age05 age06 age07 age08 age09 age10 age11 age12 age13 age14 age15 age16;
array SE30 SE3001 SE3002 SE3003 SE3004 SE3005 SE3006 SE3007 SE3008 SE3009 SE3010 SE3011 SE3012
      SE3013 SE3014 SE3015 SE3016;
array SE31 SE3101 SE3102 SE3103 SE3104 SE3105 SE3106 SE3107 SE3108 SE3109 SE3110 SE3111 SE3112
      SE3113 SE3114 SE3115 SE3116;

do I=1 to 16;
  if SE28Y(I) eq -4 then age(I)=-4;
```

```

if SE28Y(I) eq -3 then age(I)=-3;
if SE28Y(I) eq -2 then age(I)=-2;
if SE28Y(I) eq -1 then age(I)=-1;
if SE28Y(I) gt SE1Y then age(I)=-2;
if SE28Y(I) le SE1Y and SE28Y(I) gt 0 and SE1Y gt 0 then age(I)=(SE1Y-SE28Y(I));
if age(I) ge 0 and SE28M(I) gt SE1M and SE1M gt 0 and SE28M(I) gt 0 then age(I)=(SE1Y-SE28Y(I)-1);
if age(I) ge 0 and SE28M(I) lt SE1M and SE28M(I) gt 0 and SE1M gt 0 then age(I)=(SE1Y-SE28Y(I));
if age(I) ge 0 and SE28M(I) eq SE1M and SE28D(I) gt SE1D and SE1D gt 0 and SE1D gt 0 then age(I)=(SE1Y-SE28Y(I)-1);
if age(I) ge 0 and SE28M(I) eq SE1M and SE28D(I) le SE1D and SE28D(I) gt 0 and SE1D gt 0 then
    age(I)=(SE1Y-SE28Y(I));
if SE31(I) ge 0 then age(I)=SE31(I);
end;

array dummy1 dummy101 dummy102 dummy103 dummy104 dummy105 dummy106 dummy107 dummy108
    dummy109 dummy110 dummy111 dummy112 dummy113 dummy114 dummy115 dummy116;
array dummy2 dummy201 dummy202 dummy203 dummy204 dummy205 dummy206 dummy207 dummy208
    dummy209 dummy210 dummy211 dummy212 dummy213 dummy214 dummy215 dummy216;
array dummy3 dummy301 dummy302 dummy303 dummy304 dummy305 dummy306 dummy307 dummy308
    dummy309 dummy310 dummy311 dummy312 dummy313 dummy314 dummy315 dummy316;
array SE32 SE3201 SE3202 SE3203 SE3204 SE3205 SE3206 SE3207 SE3208 SE3209 SE3210 SE3211 SE3212
    SE3213 SE3214 SE3215 SE3216;
array SE34 SE3401 SE3402 SE3403 SE3404 SE3405 SE3406 SE3407 SE3408 SE3409 SE3410 SE3411 SE3412
    SE3413 SE3414 SE3415 SE3416;

do I=1 to 16;
  if age(I) lt 6 and age(I) ge 0 then dummy1(I)=1; else dummy1(I)=0;
  if (age(I) lt 18 and age(I) ge 0) or SE32(I)=1 or SE34(I)=1 then dummy2(I)=1; else dummy2(I)=0;
end;

do I=1 to 16;
  if age(I) eq -1 then under6=-1;
  if age(I) eq -2 then under6=-2;
  if age(I) eq -3 then under6=-3;
  if age(I) eq -1 and SE32(I) ne 1 and SE34(I) ne 1 then under18=-1;
  if age(I) eq -2 and SE32(I) ne 1 and SE34(I) ne 1 then under18=-2;
  if age(I) eq -3 and SE32(I) ne 1 and SE34(I) ne 1 then under18=-3;
end;

do I=1 to 16;
under6=dummy101+dummy102+dummy103+dummy104+dummy105+dummy106+dummy107+
    dummy108+dummy109+dummy110+dummy111+dummy112+dummy113+dummy114+
    dummy115+dummy116;
under18=dummy201+dummy202+dummy203+dummy204+dummy205+dummy206+dummy207+
    dummy208+dummy209+dummy210+dummy211+dummy212+dummy213+dummy214+
    dummy215+dummy216;
end;

do I=1 to 16;
  if (SE31(I) eq -1 or SE31(I) eq -2 or SE31(I) eq -3) and (SE28M(I) eq -1 or SE28M(I) eq -2 or SE28M(I) eq -3)
    then A=1; else A=0;
end;

/* Some of the records are hand edited. */
if pubid in (1281, 1954) then do;
  hhmember=5; under6=1; under18=3; end;

```

Appendix 3: Family Background Variable Creation

```
if pubid in (1947, 2394, 3841, 3867, 3868, 4009, 4033, 4032, 4921, 4922, 6349, 6350) then do;  
    hhmember=4; under6=0; under18=2; end;  
  
if pubid in (1949, 6691) then do;  
    hhmember=2; under6=0; under18=1; end;  
  
if pubid in (1959, 8850) then do;  
    hhmember=3; under6=0; under18=1; end;  
  
if pubid=2284 then do;  
    hhmember=6; under6=1; under18=3; end;  
  
if pubid=4027 then do;  
    hhmember=4; under6=0; under18=1; end;  
  
if pubid in (4053, 6334, 6335) then do;  
    hhmember=5; under6=0; under18=3; end;  
  
if pubid=6803 or pubid=6804 then do;  
    hhmember=8; under6=1; under18=5; end;  
  
if pubid=8818 or pubid=8819 or pubid=8820 or pubid=8821 then do;  
    hhmember=6; under6=0; under18=4; end;  
  
endsas;
```

YOUTH'S MARITAL STATUS AS OF THE SURVEY DATE

Variables Created: CV_MARSTAT
CV_MARSTAT_COLLAPSED

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
M700	YMAR-700	M540011, 12	YMAR-5400.01.01, .02
M900	YMAR-900	M57001	YMAR-570001
M1200	YMAR-1200	M730011	YMAR-7300.01.01
M3100Y1, Y2	YMAR-3100.01_Y, .02_Y	M730021	YMAR-7300.02.01
M45001, 02	YMAR-4500.01, .02	PUBID	PUBID
M47001, 02	YMAR-4700.01, .02		

Codes for Created Variable

Marital/Cohabitation Status	Collapsed Marital Status
1 = never married, cohabiting	0 = never married
2 = never married, not cohabiting	1 = married
3 = married, spouse present	2 = separated
4 = married, spouse absent	3 = divorced
5 = separated, cohabiting	4 = widowed
6 = separated, not cohabiting	
7 = divorced, cohabiting	
8 = divorced, not cohabiting	
9 = widowed, cohabiting	
10 = widowed, not cohabiting	

This program creates two marital status/cohabitation status variables for respondents age 16 and older. Other respondents are valid skips (-4). Note that later partners take precedence over earlier partners.

***** COLLAPSED MARITAL STATUS VARIABLE *****.

*Initialize to not married.

COMPUTE CMARSTAT=0.

*If youth is less than 16 years old then valid skip.

IF (M700=1) CMARSTAT=-4.

*Establish whether married to partner 1. To be married R must report a partner and a valid marriage date.

IF (M1200>0 AND M45001=1 AND M3100Y1>0) MARRY1=1.

IF (M1200>0 AND M45001=0 AND (M540011=1 AND M5700Y1>0)) MARRY1=1.

*Not legally married at start (M45001=1). Relationship ended - partner 1, period 1.

IF (MARRY1=1 AND M47001=0) CMARSTAT=1.

IF (MARRY1=1 AND M47001=0 AND M540011=3 AND M5600Y11>0) CMARSTAT=2.

IF (MARRY1=1 AND M47001=0 AND M540011=4 AND M5600Y11>0) CMARSTAT=3.

IF (MARRY1=1 AND M47001=0 AND M114001=5 AND M5600Y11>0) CMARSTAT=4.

*Not legally married at start (M45001=1). Relationship ended - partner 1, period 2.

IF (MARRY1=1 AND M47001=0 AND M540012=3 AND M5600Y12>0) CMARSTAT=2.

IF (MARRY1=1 AND M47001=0 AND M540012=4 AND M5600Y12>0) CMARSTAT=3.

IF (MARRY1=1 AND M47001=0 AND M114002=5 AND M5600Y12>0) CMARSTAT=4.

*If living together continuously, no changes - partner 1, period 1.

IF (MARRY1=1 AND M47001=1 AND M730011=0) CMARSTAT=1.

*Establish whether married to second partner.

IF (M1200>0 AND M45002=1 AND M3100Y2>0) MARRY2=1.

*If living together continuously, no changes - partner 2, period 1.

IF (MARRY2=1 AND M47002=1 AND M730021=0) CMARSTAT=1.

*If can't determine beginning status (married or not).

IF (M45001=-1 OR M45002=-1 OR M3100Y1=-1 OR M3100Y2=-1) CMARSTAT=-1.

Appendix 3: Family Background Variable Creation

```

IF (M45001=-2 OR M45002=-2 OR M3100Y1=-2 OR M3100Y2=-2) CMARSTAT=-2.
IF (M45001=-3 OR M45002=-3 OR M3100Y1=-3 OR M3100Y2=-3) CMARSTAT=-3.

*If can't determine changes in status.
IF (CMARSTAT>0 AND (M47001=-1 OR M47002=-1 OR M540011=-1)) CMARSTAT=-1.
IF (CMARSTAT>0 AND (M540012=-1 OR M5600Y11=-1 OR M5600Y12=-1 )) CMARSTAT=-1.
IF (CMARSTAT>0 AND (M5700Y1=-1 OR M730011=-1 OR M730021=-1)) CMARSTAT=-1.
IF (CMARSTAT>0 AND (M114001=-1 OR M114002=-1)) CMARSTAT=-1.

*If can't determine changes in status.
IF (CMARSTAT>0 AND (M47001=-2 OR M47002=-2 OR M540011=-2)) CMARSTAT=-2.
IF (CMARSTAT>0 AND (M540012=-2 OR M5600Y11=-2 OR M5600Y12=-2 )) CMARSTAT=-2.
IF (CMARSTAT>0 AND (M5700Y1=-2 OR M730011=-2 OR M730021=-2)) CMARSTAT=-2.
IF (CMARSTAT>0 AND (M114001=-2 OR M114002=-2)) CMARSTAT=-2.

*If can't determine changes in status.
IF (CMARSTAT>0 AND (M47001=-3 OR M47002=-3 OR M540011=-3)) CMARSTAT=-3.
IF (CMARSTAT>0 AND (M540012=-3 OR M5600Y11=-3 OR M5600Y12=-3 )) CMARSTAT=-3.
IF (CMARSTAT>0 AND (M5700Y1=-3 OR M730011=-3 OR M730021=-3)) CMARSTAT=-3.
IF (CMARSTAT>0 AND (M114001=-3 OR M114002=-3)) CMARSTAT=-3.

*on roster from screener but R reports -1/-2 for # partners, don't include.
IF (M900=1 AND (M1200=-1 OR M1200=-2)) CMARSTAT=0.

```

***** YOUTH'S MARITAL OR COHABITATION STATUS AS OF THE SURVEY DATE *****.

/Initialize to never married, partner not present.*

```

COMPUTE MARCOHAB=2.

IF (CMARSTAT=0 AND M47001=1) MARCOHAB=1.
IF (CMARSTAT=0 AND M47001=0) MARCOHAB=2.
IF (CMARSTAT=1 AND M47001=1) MARCOHAB=3.
IF (CMARSTAT=1 AND M47001=0) MARCOHAB=4.
IF (CMARSTAT=2 AND M47001=1) MARCOHAB=5.
IF (CMARSTAT=2 AND M47001=0) MARCOHAB=6.
IF (CMARSTAT=3 AND M47001=1) MARCOHAB=7.
IF (CMARSTAT=3 AND M47001=0) MARCOHAB=8.
IF (CMARSTAT=4 AND M47001=1) MARCOHAB=9.
IF (CMARSTAT=4 AND M47001=0) MARCOHAB=10.
IF (CMARSTAT=0 AND M47002=1) MARCOHAB=1.
IF (CMARSTAT=0 AND M47002=0) MARCOHAB=2.
IF (CMARSTAT=1 AND M47002=1) MARCOHAB=3.
IF (CMARSTAT=1 AND M47002=0) MARCOHAB=4.
IF (CMARSTAT=2 AND M47002=1) MARCOHAB=5.
IF (CMARSTAT=2 AND M47002=0) MARCOHAB=6.
IF (CMARSTAT=3 AND M47002=1) MARCOHAB=7.
IF (CMARSTAT=3 AND M47002=0) MARCOHAB=8.
IF (CMARSTAT=4 AND M47002=1) MARCOHAB=9.
IF (CMARSTAT=4 AND M47002=0) MARCOHAB=10.

*Do never married, partner 1 present.*  

*Do never married, partner 1 not present.*  

*Do married, spouse 1 present.*  

*Do married, spouse 1 not present.*  

*Do separated, partner 1 present.*  

*Do separated, partner 1 not present.*  

*Do divorced, partner 1 present.*  

*Do divorced, partner 1 not present.*  

*Do widowed, partner 1 present.*  

*Do widowed, partner 1 not present.*  

*Do never married, partner 2 present.*  

*Do never married, partner 2 not present.*  

*Do married, spouse 2 present.*  

*Do married, spouse 2 not present.*  

*Do separated, partner 2 present.*  

*Do separated, partner 2 not present.*  

*Do divorced, partner 2 present.*  

*Do divorced, partner 2 not present.*  

*Do widowed, partner 2 present.*  

*Do widowed, partner 2 not present.*
```

*Do all else.

```

IF (CMARSTAT=-1 OR M47001=-1 OR M47002=-1) MARCOHAB=-1.
IF (CMARSTAT=-2 OR M47001=-2 OR M47002=-2) MARCOHAB=-2.
IF (CMARSTAT=-3 OR M47001=-3 OR M47001=-3) MARCOHAB=-3.
IF (CMARSTAT=-4) MARCOHAB=-4.

```

*On roster from screener but R reports -1/-2 for # partners, don't include.

```
IF (M900=1 AND (M1200=-1 OR M1200=-2)) MARCOHAB=2.
```

*To correct for invalid skips.

```
IF (M700=-3) CMARSTAT=-3 and MARCOHAB=-3.
```

EXECUTE.

YOUTH'S RELATIONSHIP TO HOUSEHOLD PARENT FIGURE(S)

Variables Created: CV_HH_REL_BIRTH

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
pubid	PUBID	REL13_1-16	HHI2_REL13.02-.16
saq13	YSAQ-013	REL14_1-16	HHI2_REL14.01-.16
saq14	YSAQ-014	REL15_1-16	HHI2_REL15.01-.16
saq34	YSAQ-034	REL16_1-16	HHI2_REL16.01-.16
saq35	YSAQ-035	REL2_1-16	HHI2_REL2.01-.16
yid	YOUTH_ID.01	REL3_1-16	HHI2_REL3.01-.16
YMOMID	YOUTH_MOMID.01	REL4_1-16	HHI2_REL4.01-.16
YNRMOMID	YOUTH_NRMOMID.01	REL5_1-16	HHI2_REL5.01-.16
birthd, m, y	KEY!BDATE_D, _M, _Y	REL6_1-16	HHI2_REL6.01-.16
yAGE	KEY!AGE	REL7_1-16	HHI2_REL7.01-.16
PNRMOMID	PARYOUTH_MOMID	REL8_1-16	HHI2_REL8.01-.16
ScrD, M, Y	SE-1_D, _M, _Y	REL9_1-16	HHI2_REL9.01-.16
SN701-05	SN-70.01-.05	NAGE1-23	NONHHI AGE.01-.23
SN761-3	SN-76.01-.03	NREL1-21	NONHHI_RELATION.01-.21
SN791-3	SN-79.01-.03	PI017	PINF-017
AGE1-16	HHI2_AGE.01-.16	PI018	PINF-018
DOBD1-15	HHI2_DOB.01_D-.15_D	PI020	PINF-020
DOBM1-15	HHI2_DOB.01_M-.15_M	PI025	PINF-025
DOBY1-15	HHI2_DOB.01_Y-.15_Y	PI0455	PINF-045_005
REL1_1-16	HHI2_REL1.01-.16	PI058D, M, Y	PINF-058_D, _M, _Y
REL10_1-16	HHI2_REL10.01-10.16	gender1-16	HHI2_SEX.01-.16
REL11_1-16	HHI2_REL11.01-11.16	ngend1-23	NONHHI SEX.01-.23
REL12_1-16	HHI2_REL12.01-.16		

Codes for Created Variable

- | | |
|------------------------------------|---------------------------------|
| 1 = Both biological parents | 6 = Adoptive parent(s) |
| 2 = Two parents, biological mother | 7 = Foster parent(s) |
| 3 = Two parents, biological father | 8 = No parents, grandparents |
| 4 = Biological mother only | 9 = No parents, other relatives |
| 5 = Biological father only | 10 = Anything else |

This program creates a variable identifying the youth's relationship to the primary adults in the household.

```

array rela1 (i) rel1_1-rel1_19;
array rela2 (i) rel2_1-rel2_19;
array rela3 (i) rel3_1-rel3_19;
array rela4 (i) rel4_1-rel4_19;
array rela5 (i) rel5_1-rel5_19;
array rela6 (i) rel6_1-rel6_19;
array rela7 (i) rel7_1-rel7_19;
array rela8 (i) rel8_1-rel8_19;
array rela9 (i) rel9_1-rel9_19;
array rela10 (i) rel10_1-rel10_19;
array rela11 (i) rel11_1-rel11_19;
array rela12 (i) rel12_1-rel12_19;
array rela13 (i) rel13_1-rel13_19;
array rela14 (i) rel14_1-rel14_19;
array rela15 (i) rel15_1-rel15_19;
array rela16 (i) rel16_1-rel16_19;
array rela17 (i) rel17_1-rel17_19;

array rela18 (i) rel18_1-rel18_19;
array rela19 (i) rel19_1-rel19_19;
array parid (i) parid1-parid19;

momid=0;
domid=0;
adopdad=0;
admom=0;
fostma=0;
fostda=0;
stepma=0;
stepda=0;
husb=0;
wife=0;
grand=0;
relat=0;
nonrel=0;

```

```

do i=1 to 19;
if yid=1 then do;
  if rela1>48 and rela1<57 then do;
    parid=1;
  end;
  if rela1=47 or rela1=48 then do;
    grand=1;
  end;
  if rela1=1 or rela1=2 or (rela1>12 and rela1<19)
    then do;
    relat=1;
  end;
  if rela1=74 or rela1=78 or (rela1>81 and rela1<85)
    or rela1=86 or rela1=87 then do;
    relat=1;
  end;
  if rela1=59 or rela1=60 or rela1=68 or rela1=69 or
    rela1=85 then do;
    nonrel=1;
  end;
  if rela1=-1 or rela1=-2 or rela1=-3 then do;
    invalid=-3;
  end;
end;

if yid=2 then do;
  if rela2>48 and rela2<57 then do;
    parid=1;
  end;
  if rela2=47 or rela2=48 then do;
    grand=1;
  end;
  if rela2=1 or rela2=2 then do;
    relat=1;
  end;
  if rela2=74 or rela2=78 or (rela2>81 and rela2<85)
    or rela2=86 or rela2=87 then do;
    relat=1;
  end;
  if rela2=59 or rela2=60 or rela2=68 or rela2=69 or
    rela2=85 then do;
    nonrel=1;
  end;
  if rela2=-1 or rela2=-2 or rela2=-3 then do;
    invalid=-3;
  end;
end;

if yid=3 then do;
  if rela3>48 and rela3<57 then do;
    parid=1;
  end;
  if rela3=47 or rela3=48 then do;
    grand=1;
  end;
  if rela3=1 or rela3=2 then do;
    relat=1;
  end;
  if rela3=74 or rela3=78 or (rela3>81 and rela3<85)
    or rela3=86 or rela3=87 then do;
    relat=1;
  end;
  if rela3=59 or rela3=60 or rela3=68 or rela3=69 or
    rela3=85 then do;
    nonrel=1;
  end;
  if rela3=-1 or rela3=-2 or rela3=-3 then do;
    invalid=-3;
  end;
end;

if yid=4 then do;
  if rela4>48 and rela4<57 then do;
    parid=1;
  end;
  if rela4=47 or rela4=48 then do;
    grand=1;
  end;
  if rela4=1 or rela4=2 then do;
    relat=1;
  end;
  if rela4=74 or rela4=78 or (rela4>81 and rela4<85)
    or rela4=86 or rela4=87 then do;
    relat=1;
  end;
  if rela4=59 or rela4=60 or rela4=68 or rela4=69 or
    rela4=85 then do;
    nonrel=1;
  end;
  if rela4=-1 or rela4=-2 or rela4=-3 then do;
    invalid=-3;
  end;
end;

if yid=5 then do;
  if rela5>48 and rela5<57 then do;
    parid=1;
  end;
  if rela5=47 or rela5=48 then do;
    grand=1;
  end;
  if rela5=1 or rela5=2 then do;
    relat=1;
  end;
  if rela5=74 or rela5=78 or (rela5>81 and rela5<85)
    or rela5=86 or rela5=87 then do;
    relat=1;
  end;
  if rela5=59 or rela5=60 or rela5=68 or rela5=69 or
    rela5=85 then do;
    nonrel=1;
  end;
  if rela5=-1 or rela5=-2 or rela5=-3 then do;
    invalid=-3;
  end;
end;

```

```

        invalid=-3;
    end;
end;
end;

do i=1 to 19;
if yid=1 then do;
    if parid1=1 and rel1_1=5 then admom=1;
    if parid2=1 and rel2_1=5 then admom=2;
    if parid3=1 and rel3_1=5 then admom=3;
    if parid4=1 and rel4_1=5 then admom=4;
    if parid5=1 and rel5_1=5 then admom=5;
    if parid6=1 and rel6_1=5 then admom=6;
    if parid7=1 and rel7_1=5 then admom=7;
    if parid8=1 and rel8_1=5 then admom=8;
    if parid9=1 and rel9_1=5 then admom=9;
    if parid10=1 and rel10_1=5 then admom=10;
    if parid11=1 and rel11_1=5 then admom=11;
    if parid12=1 and rel12_1=5 then admom=12;
    if parid13=1 and rel13_1=5 then admom=13;
    if parid14=1 and rel14_1=5 then admom=14;
    if parid15=1 and rel15_1=5 then admom=15;
    if parid16=1 and rel16_1=5 then admom=16;
    if parid17=1 and rel17_1=5 then admom=17;
    if parid18=1 and rel18_1=5 then admom=18;
    if parid19=1 and rel19_1=5 then admom=19;
end;

if yid=2 then do;
    if parid1=1 and rel1_2=5 then admom=1;
    /* etc. through rel19_2=5 and admom=19 */
end;
if yid=5 then do;
    if parid1=1 and rel1_5=5 then admom=1;
    /* etc. through rel19_5=5 and admom=19 */
end;
if yid=4 then do;
    if parid1=1 and rel1_4=5 then admom=1;
    /* etc. through rel19_4=5 and admom=19 */
end;
if yid=5 then do;
    if parid1=1 and rel1_5=5 then admom=1;
    /* etc. through rel19_5=5 and admom=19 */
end;
if yid=1 to 19;
if yid=1 then do;
    if parid1=1 and rel1_1=6 then adopdad=1;
    /* etc. through rel19_1=6 and adopdad=19 */
end;
if yid=2 then do;
    if parid1=1 and rel1_2=6 then adopdad=1;
    /* etc. through rel19_2=6 and adopdad=19 */
end;
if yid=3 then do;
    if parid1=1 and rel1_3=6 then adopdad=1;
    /* etc. through rel19_3=6 and adopdad=19 */
end;
if yid=4 then do;
    if parid1=1 and rel1_4=6 then adopdad=1;
    /* etc. through rel19_4=6 and adopdad=19 */
end;
if yid=5 then do;
    if parid1=1 and rel1_5=6 then adopdad=1;
    /* etc. through rel19_5=6 and adopdad=19 */
end;
do i=1 to 19;
if yid=1 then do;
    if parid1=1 and rel1_1=3 then momid=1;
    /* etc. through rel19_1=3 and momid=19 */
end;
if yid=2 then do;
    if parid1=1 and rel1_2=3 then momid=1;
    /* etc. through rel19_2=3 and momid=19 */
end;
if yid=3 then do;
    if parid1=1 and rel1_3=3 then momid=1;
    /* etc. through rel19_3=3 and momid=19 */
end;
if yid=4 then do;
    if parid1=1 and rel1_4=3 then momid=1;
    /* etc. through rel19_4=3 and momid=19 */
end;
if yid=5 then do;
    if parid1=1 and rel1_5=3 then momid=1;
    /* etc. through rel19_5=3 and momid=19 */
end;
do i=1 to 19;
if yid=1 then do;
    if parid1=1 and rel1_1=4 then domid=1;
    /* etc. through rel19_1=4 and domid=19 */
end;
if yid=2 then do;
    if parid1=1 and rel1_2=4 then domid=1;
    /* etc. through rel19_2=4 and domid=19 */
end;
if yid=3 then do;
    if parid1=1 and rel1_3=4 then domid=1;
    /* etc. through rel19_3=4 and domid=19 */
end;
if yid=4 then do;
    if parid1=1 and rel1_4=4 then domid=1;
    /* etc. through rel19_4=4 and domid=19 */
end;
if yid=5 then do;
    if parid1=1 and rel1_5=4 then domid=1;
    /* etc. through rel19_5=4 and domid=19 */
end;
do i=1 to 19;

```

```

if yid=1 then do;
    if parid1=1 and rel1_1=9 then fostma=1;
        /* etc. through rel19_1=9 and fostma=19 */
    end;
if yid=2 then do;
    if parid1=1 and rel1_2=9 then fostma=1;
        /* etc. through rel19_2=9 and fostma=19 */
    end;
if yid=3 then do;
    if parid1=1 and rel1_3=9 then fostma=1;
        /* etc. through rel19_3=9 and fostma=19 */
    end;
if yid=4 then do;
    if parid1=1 and rel1_4=9 then fostma=1;
        /* etc. through rel19_4=9 and fostma=19 */
    end;
if yid=5 then do;
    if parid1=1 and rel1_5=9 then fostma=1;
        /* etc. through rel19_5=9 and fostma=19 */
    end;
end;

do i=1 to 19;
    if yid=1 then do;
        if parid1=1 and rel1_1=10 then fostda=1;
            /* etc. through rel19_1=10 and fostda=19 */
        end;
    if yid=2 then do;
        if parid1=1 and rel1_2=10 then fostda=1;
            /* etc. through rel19_2=10 and fostda=19 */
        end;
    if yid=3 then do;
        if parid1=1 and rel1_3=10 then fostda=1;
            /* etc. through rel19_3=10 and fostda=19 */
        end;
    if yid=4 then do;
        if parid1=1 and rel1_4=10 then fostda=1;
            /* etc. through rel19_4=10 and fostda=19 */
        end;
    if yid=5 then do;
        if parid1=1 and rel1_5=10 then fostda=1;
            /* etc. through rel19_5=10 and fostda=19 */
        end;
    end;

do i=1 to 19;
    if yid=1 then do;
        if parid1=1 and rel1_1=7 then stepma=1;
            /* etc. through rel19_1=7 and stepma=19 */
        end;
    if yid=2 then do;
        if parid1=1 and rel1_2=7 then stepma=1;
            /* etc. through rel19_2=7 and stepma=19 */
        end;
    if yid=3 then do;
        if parid1=1 and rel1_3=7 then stepma=1;
            /* etc. through rel19_3=7 and stepma=19 */
        end;
    end;

/* etc. through rel19_3=7 and stepma=19 */
end;
if yid=4 then do;
    if parid1=1 and rel1_4=7 then stepma=1;
        /* etc. through rel19_4=7 and stepma=19 */
    end;
if yid=5 then do;
    if parid1=1 and rel1_5=7 then stepma=1;
        /* etc. through rel19_5=7 and stepma=19 */
    end;
end;

do i=1 to 19;
    if yid=1 then do;
        if parid1=1 and rel1_1=8 then stepda=1;
            /* etc. through rel19_1=8 and stepda=19 */
        end;
    if yid=2 then do;
        if parid1=1 and rel1_2=8 then stepda=1;
            /* etc. through rel19_2=8 and stepda=19 */
        end;
    if yid=3 then do;
        if parid1=1 and rel1_3=8 then stepda=1;
            /* etc. through rel19_3=8 and stepda=19 */
        end;
    if yid=4 then do;
        if parid1=1 and rel1_4=8 then stepda=1;
            /* etc. through rel19_4=8 and stepda=19 */
        end;
    if yid=5 then do;
        if parid1=1 and rel1_5=8 then stepda=1;
            /* etc. through rel19_5=8 and stepda=19 */
        end;
    end;

do i=1 to 19;
    if momid>0 then do;
        if momid=1 and rela1=1 then husb=1;
        if momid=2 and rela2=1 then husb=2;
        if momid=3 and rela3=1 then husb=3;
        if momid=4 and rela4=1 then husb=4;
        if momid=5 and rela5=1 then husb=5;
        if momid=6 and rela6=1 then husb=6;
        if momid=7 and rela7=1 then husb=7;
        if momid=8 and rela8=1 then husb=8;
        if momid=9 and rela9=1 then husb=9;
        if momid=10 and rela10=1 then husb=10;
        if momid=11 and rela11=1 then husb=11;
        if momid=12 and rela12=1 then husb=12;
        if momid=13 and rela13=1 then husb=13;
        if momid=14 and rela14=1 then husb=14;
        if momid=15 and rela15=1 then husb=15;
        if momid=16 and rela16=1 then husb=16;
        if momid=17 and rela17=1 then husb=17;
        if momid=18 and rela18=1 then husb=18;
        if momid=19 and rela19=1 then husb=19;
    end;
end;

```

Appendix 3: Family Background Variable Creation

```
end;  
  
do i=1 to 19;  
  if domid>0 then do;  
    if domid=1 and rela1=2 then wife=1;  
    if domid=2 and rela2=2 then wife=2;  
    if domid=3 and rela3=2 then wife=3;  
    if domid=4 and rela4=2 then wife=4;  
    if domid=5 and rela5=2 then wife=5;  
    if domid=6 and rela6=2 then wife=6;  
    if domid=7 and rela7=2 then wife=7;  
    if domid=8 and rela8=2 then wife=8;  
    if domid=9 and rela9=2 then wife=9;  
    if domid=10 and rela10=2 then wife=10;  
    if domid=11 and rela11=2 then wife=11;  
    if domid=12 and rela12=2 then wife=12;  
    if domid=13 and rela13=2 then wife=13;  
    if domid=14 and rela14=2 then wife=14;  
    if domid=15 and rela15=2 then wife=15;  
    if domid=16 and rela16=2 then wife=16;  
    if domid=17 and rela17=2 then wife=17;  
    if domid=18 and rela18=2 then wife=18;  
    if domid=19 and rela19=2 then wife=19;  
  end;  
end;  
  
rel=10;  
  
if invalid=-3 then do;  
  rel=-3; end;  
  
if nonrel>0 then do;  
  rel=10; end;  
  
  if relat>0 then do;  
    rel=9; end;  
  
  if grand>0 and momid=0 and domid=0 then do;  
    rel=8; end;  
  
  if fostda>0 or fostma>0 then do;  
    rel=7; end;  
  
  if admom>0 or adopdad>0 then do;  
    rel=6; end;  
  
  if domid>0 and wife=0 and momid=0 then do;  
    rel=5; end;  
  
  if momid>0 and husb=0 and domid=0 then do;  
    rel=4; end;  
  
  if domid>0 and momid=0 then do;  
    if wife>0 or admom>0 or stepma>0 then rel=3; end;  
  
  if momid>0 and domid=0 then do;  
    if husb>0 or adopdad>0 or stepda>0 then rel=2;  
    end;  
  
  if momid>0 and domid>0 then do;  
    both=1;  
    rel=1; end;  
  
*hand edits;  
if pubid in (287, 288, 289, 1470, 4127, 5345, 5346)  
  then rel=1;  
  
endsas;
```

YOUTH'S FERTILITY AND CHILD STATUS

Variables Created:	CV_CHILD_BIRTH_DATE.xx_M CV_CHILD_DEATH_DATE.xx_M CV_CHILD_BIRTH_MONTH.xx CV_CHILD_STATUS.xx CV_BIO_CHILD_HH	CV_CHILD_BIRTH_DATE.xx_Y CV_CHILD_DEATH_DATE.xx_Y CV_CHILD_DEATH_MONTH.xx CV_BIO_CHILD_NR
---------------------------	--	--

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
F300	YFER-300	BDAYD1-D4	BIOCHILD_BDATE.01_D-.04_D
F7200M1, Y1	YFER-7200M1, Y1	BDAYM1-M4	BIOCHILD_BDATE.01_M-.04_M
F7200M2, Y2	YFER-7200M2, Y2	BDAYY1-Y4	BIOCHILD_BDATE.01_Y-.04_Y
F126001-04	YFER-12600.01-.04	BDEAD1, 2	BIOCHILD_DEAD.01, .02
F128001, 02	YFER-12800.01, .02	RESHM1-4	BIOCHILD_RESIDE.01-04
F129001-04	YFER-12900.01-.04	PUBID	PUBID

Codes for Created Variables

Date of birth and death variables

Date variables are presented as both the actual month and year and the month number in a continuous month scheme.

Status variables

- 1 Adopted
- 2 Deceased
- 3 Non-resident, foster care
- 4 Non-resident, not adopted or in foster care
- 5 Resident

This program creates a number of variables describing the youth's fertility and the current status of the youth's children. For more information on the continuous month system, see appendix 7 in this document.

/* First, create a variable indicating the DOBM(I) and DOBY(I) for each biological child. Information is taken from the fertility roster (CHIL) and the fertility section of the youth survey (YFER). */

```

array BDAYM[4] BDAYM1-BDAYM4;
array BDAYY[4] BDAYY1-BDAYY4;
array DOBM[4] DOBM1-DOBM4 ;
array DOBY[4] DOBY1-DOBY4 ;

do I=1 to 4;
DOBM(I)=-4;
if BDAYM(I) eq -3 or F300 eq -3 then DOBM(I)=-3;
if BDAYM(I) eq -1 then DOBM(I)=-1;
if BDAYM(I) eq -2 then DOBM(I)=-2;
if BDAYM(I) gt 0 then DOBM(I)=BDAYM(I);
end;

a=0;
do i=1 to 3 while(DOBM(i) eq .);
a=a+1;
end;
if a ne 0 then do;
do i=1 to 4;
if (i+a) le 4 then do;
DOBM(i)=DOBM(i+a);
DOBM(i+a)=.;
end;
end;

```

Appendix 3: Family Background Variable Creation

end;

```
do I=1 to 4;  
DOBY(I)=-4;  
if BDAYY(I) eq -3 or F300 eq -3 then DOBY(I)=-3;  
if BDAYY(I) eq -1 then DOBY(I)=-1;  
if BDAYY(I) eq -2 then DOBY(I)=-2;  
if BDAYY(I) gt 0 then DOBY(I)=BDAYY(I);  
end;
```

```
b=0;  
do i=1 to 3 while(DOBY(i) eq .);  
b=b+1;  
end;  
if b ne 0 then do;  
do i=1 to 4;  
if (i+b) le 4 then do;  
DOBY(i)=DOBY(i+b);  
DOBY(i+b)=.;  
end;  
end;  
end;
```

/ Second, create a continuous month scheme variable for the month of birth of the children using the formula:
(12*(DOBY(I)-1980)+DOBM(I)) */*

```
array MOB[4] MOB1-MOB4 ;
```

```
do I=1 to 4;  
MOB(I)=-4;  
if DOBM(I) eq -3 or DOBY(I) eq -3 or F300 eq -3 then MOB(I)=-3;  
if DOBM(I) eq -1 or DOBY(I) eq -1 then MOB(I)=-1;  
if DOBM(I) eq -2 or DOBY(I) eq -2 then MOB(I)=-2;  
if DOBM(I) gt 0 and DOBY(I) ge 1980 then MOB(I)=12*(DOBY(I)-1980)+DOBM(I);  
end;  
c=0;  
do i=1 to 3 while(MOB(i) eq .);  
c=c+1;  
end;  
if c ne 0 then do;  
do i=1 to 4;  
if (i+c) le 4 then do;  
MOB(i)=MOB(i+c);  
MOB(i+c)=.;  
end;  
end;  
end;
```

/ Third, create an actual date variable for the date of death of the youth's children. Only the DOD of children not included in the roster is available, the ones for any children listed in the roster and possibly dead is not provided. */*

```
array BDEAD[2] BDEAD1-BDEAD2;  
array F7200M[2] F7200M1-F7200M2;  
array F7200Y[2] F7200Y1-F7200Y2;  
array DODM[2] DODM1-DODM2;
```

```

array DODY[2] DODY1-DODY2;

do I=1 to 2;
    if BDEAD(I) eq 1 then do;
        if F7200M(I) eq -4 then DODM(I)=-4;
        if F7200M(I) eq -3 or F300 eq -3 then DODM(I)=-3;
        if F7200M(I) eq -1 then DODM(I)=-1;
        if F7200M(I) eq -2 then DODM(I)=-2;
        if F7200M(I) gt 0 then DODM(I)=F7200M(I);
    end;
    else DODM(I)=-4;
end;

d=0;
do i=1 to 1 while(DODM(i) eq .);
    d=d+1;
end;
if d ne 0 then do;
    do i=1 to 2;
        if (i+d) le 2 then do;
            DODM(i)=DODM(i+d);
            DODM(i+d)=.;
        end;
    end;
end;

do I=1 to 2;
    if BDEAD(I) eq 1 then do;
        if F7200Y(I) eq -4 then DODY(I)=-4;
        if F7200Y(I) eq -3 or F300 eq -3 then DODY(I)=-3;
        if F7200Y(I) eq -1 then DODY(I)=-1;
        if F7200Y(I) eq -2 then DODY(I)=-2;
        if F7200Y(I) gt 0 then DODY(I)=F7200Y(I);
    end;
    else DODY(I)=-4;
end;

e=0;
do i=1 to 1 while(DODY(i) eq .);
    e=e+1;
end;
if e ne 0 then do;
    do i=1 to 2;
        if (i+e) le 2 then do;
            DODY(i)=DODY(i+e);
            DODY(i+e)=.;
        end;
    end;
end;

```

/* Fourth, create a continuous month scheme variable forthe month of death of the children using the formula:
 $(12*(DODY(I)-1980)+DODM(I))$ */

```

array MOD[2] MOD1-MOD2;

do I=1 to 2;
if DODM(I) eq -4 or DODY(I) eq -4 then MOD(I)=-4;

```

Appendix 3: Family Background Variable Creation

```
if DODM(I) eq -3 or DODY(I) eq -3 or F300 eq -3 then MOD(I)=-3;
if DODM(I) eq -1 or DODY(I) eq -1 then MOD(I)=-1;
if DODM(I) eq -2 or DODY(I) eq -2 then MOD(I)=-2;
if DODM(I) gt 0 and DODY(I) ge 1980 then MOD(I)=12*(DODY(I)-1980)+DODM(I);
end;

f=0;
do i=1 to 1 while(MOD(i) eq .);
  f=f+1;
end;
if f ne 0 then do;
  do i=1 to 2;
    if (i+e) le 2 then do;
      MOD(i)=MOD(i+f);
      MOD(i+f)=.;
    end;
  end;
end;

/* Fifth, create a variable indicating the status of youth's first (second, third, fourth) child */

/* Indicating whether they live with the respondent: */

array F12600[4] F126001-F126004;
array F12800[4] F128001-F128004;
array F12900[4] F129001-F129004;
array RESHM[4] RESHM1-RESHM4;
array STATUS[4] STATUS1-STATUS4 ;

/*Initialize the STATUS variable and determine which kids live at home */

do I=1 to 4;
  STATUS(I)=-4;
  if MOB(I) eq -4 or RESHM(I) eq -4 then STATUS(I)=-4;
  if F300 eq -3 or RESHM(I) eq -3 then STATUS(I)=-3;
  if RESHM(I) eq 1 then STATUS(I)=5;
end;

/* If they are dead */
do I=1 to 2;
  if BDEAD(I) eq 1 then STATUS(I)=2;
end;

/* If they do not reside with the parents...*/
do I=1 to 4;
  /* If they were not adopted neither in foster care */
  if RESHM(I) eq 0 and BDEAD(I) ne 1 then STATUS(I)=4;
  /* If they were adopted */
  if (F12800(I)=1 or F12900(I)=4) and RESHM(I) eq 0 and BDEAD(I) ne 1 then STATUS(I)=1;
  /* If they were in foster care */
  if F12900(I)=5 and RESHM(I) eq 0 and BDEAD(I) ne 1 then STATUS(I)=3;
end;

g=0;
do i=1 to 3 while(STATUS(i) eq .);
  g=g+1;
end;
```

```
if g ne 0 then do;
do i=1 to 4;
  if (i+g) le 4 then do;
    STATUS(i)=STATUS(i+g);
    STATUS(i+g)=.:;
  end;
end;
end;

/* Sixth, the number of children ever born and residing in the household (TBIORES) */

array BIORES[4] BIORES1-BIORES4;

/*Initialize the BIORES variable and create TBIORES */

do I=1 to 4;
BIORES(I)=0;
if STATUS(I) eq 5 or RESHM(I)=1 then BIORES(I)=1;
TBIORES=BIORES1+BIORES2+BIORES3+BIORES4;
if MOB1=-4 and MOB2=-4 then TBIORES=-4;
if F300 eq -3 then TBIORES=-3;
end;

/* Seventh, the number of children ever born and not residing in the household (TBIONRES) */

array BIONRES[4] BIONRES1-BIONRES4;

do I=1 to 4;
BIONRES(I)=0;
if (STATUS(I) eq 1 or STATUS(I) eq 3 or STATUS(I) eq 4 or RESHM(I)=0) and STATUS(I) ne 2 then
BIONRES(I)=1;
TBIONRES=BIONRES1+BIONRES2+BIONRES3+BIONRES4;
if MOB1=-4 and MOB2=-4 then TBIONRES=-4;
if F300 eq -3 then TBIONRES=-3;
end;

ENDSAS;
```

MARRIAGE AND COHABITATION HISTORY

Variables Created:	CV_FIRST_MARRY_MONTH CV_FIRST_MARRY_DATE_M CV_FIRST_MARRY_DATE_Y CV_MARRIAGES_TTL	CV_FIRST_COHAB_MONTH CV_FIRST_COHAB_DATE_M CV_FIRST_COHAB_DATE_Y CV_COHAB_TTL
---------------------------	--	--

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
M5700M1, Y1	YMAR-5700.01.01_M, _Y	M1400	YMAR-1400
M3100M1, Y1	YMAR-3100.01_M, _Y	M1200	YMAR-1200
M3100M2, Y2	YMAR-3100.02_M, _Y	M45001, 02	YMAR-4500.01, .02

This program creates variables which provide the dates of the youth's first marriage and/or cohabitation in both a continuous month scheme and as actual dates (for more information on the continuous month scheme, see appendix 7 in this document). Summary variables also count the total number of marriages and cohabitutions for each youth. Note that these variables are available only for youths age 16 and older as of 12/31/96. If a respondent is cohabiting and then marries it is considered both a cohabitation and a marriage. If someone refuses or doesn't know the full date of their marriage or cohabitation, then the spell is counted in the total variables and the date variables are coded -1 or -2 as applicable.

***** MARRIAGE SECTION *****

*Compute first marriage date in continuous months since 1/1/80. Initialize to not married / valid skip.

```
COMPUTE CONTMARR = -4.  
COMPUTE MARMNTH1 = -4.  
COMPUTE MARYEAR1 = -4.
```

*<If there is a valid month (> zero) and year (> zero) marriage date, and respondent was not married when began cohabiting with partner (M45001=0) in round 1, use available marriage date data from loop 1.>

```
DO IF (M5700Y1 >= 0) AND (M5700M1 >= 0) AND (M45001=0) AND (CONTMARR = -4).  
COMPUTE CONTMARR = ((M5700Y1-1980)*12) + M5700M1.  
COMPUTE MARMNTH1 = M5700M1.  
COMPUTE MARYEAR1 = M5700Y1.  
END IF.
```

*<If they were married when started to cohabit (M45001=1), use start of cohabitation dates for the marriage date on the first loop.>

```
DO IF (M45001 = 1) AND (M3100Y1 >= 0) AND (M3100M1 >= 0) AND (CONTMARR = -4).  
COMPUTE CONTMARR = ((M3100Y1-1980)*12) + M3100M1.  
COMPUTE MARMNTH1 = M3100M1.  
COMPUTE MARYEAR1 = M3100Y1.  
END IF.
```

*<If they were married the 2nd time started to cohabit (M45002=1), use 2nd loop cohab dates for marriage date.>

```
DO IF (M45002 = 1) AND (M3100Y2 >= 0) AND (M3100M2 >= 0) AND (CONTMARR = -4).  
COMPUTE CONTMARR = ((M3100Y2-1980)*12) + M3100M2.  
COMPUTE MARMNTH1 = M3100M2.  
COMPUTE MARYEAR1 = M3100Y2.  
END IF.
```

*<Code all don't know/refused responses as appropriate>

```
IF (M5700M1=-1) AND (MARMNTH1 = -4) MARMNTH1 = -1.  
IF (M5700Y1=-1) AND (MARYEAR1 = -4) MARYEAR1 = -1.
```

```

IF (M5700M1=-2) AND (MARMNTH1 = -4) MARMNTH1 = -2.
IF (M5700Y1=-2) AND (MARYEAR1 = -4) MARYEAR1 = -2.
IF (M5700M1=-3) AND (MARMNTH1 = -4) MARMNTH1 = -3.
IF (M5700Y1=-3) AND (MARYEAR1 = -4) MARYEAR1 = -3.
IF (M3100M1=-1) AND (M45001=1) AND (MARMNTH1 = -4) MARMNTH1 = -1.
IF (M3100Y1=-1) AND (M45001=1) AND (MARYEAR1 = -4) MARYEAR1 = -1.
IF (M3100M1=-2) AND (M45001=1) AND (MARMNTH1 = -4) MARMNTH1 = -2.
IF (M3100Y1=-2) AND (M45001=1) AND (MARYEAR1 = -4) MARYEAR1 = -2.
IF (M3100M1=-3) AND (M45001=1) AND (MARMNTH1 = -4) MARMNTH1 = -3.
IF (M3100Y1=-3) AND (M45001=1) AND (MARYEAR1 = -4) MARYEAR1 = -3.
IF (M3100M2=-1) AND (M45002=1) AND (MARMNTH1 = -4) MARMNTH1 = -1.
IF (M3100Y2=-1) AND (M45002=1) AND (MARYEAR1 = -4) MARYEAR1 = -1.
IF (M3100M2=-2) AND (M45002=1) AND (MARMNTH1 = -4) MARMNTH1 = -2.
IF (M3100Y2=-2) AND (M45002=1) AND (MARYEAR1 = -4) MARYEAR1 = -2.
IF (M3100M2=-3) AND (M45002=1) AND (MARMNTH1 = -4) MARMNTH1 = -3.
IF (M3100Y2=-3) AND (M45002=1) AND (MARYEAR1 = -4) MARYEAR1 = -3.

```

* Count the total number of marriages for a respondent. Initialize to zero.

```

COMPUTE TOTMARRY = 0.
COUNT TOTMARRY = MARYEAR1 (-2, -1, 1 THRU HIGHEST).
IF (M700=1) TOTMARRY=-4.
IF (M700=-3) TOTMARRY=-3.
IF (M700=-3) MARMNTH1=-3.
IF (M700=-3) MARYEAR1=-3.
IF (M700=-3) CONTMARR=-3.

```

***** COHABITATION SECTION *****

*Compute the youth's first cohabitation in months from 1/1/80. Initialize to not cohab/valid skip.

```

COMPUTE CONTCOHB = -4.
COMPUTE CHBMNTH1 = -4.
COMPUTE CHBYEAR1 = -4.
COMPUTE CHBMNTH2 = -4.
COMPUTE CHBYEAR2 = -4.

```

*<If there is a valid month and year cohabitation date in round 1 use it.>

```

DO IF (M3100Y1 > 0) AND (M3100M1>0) AND (M45001=0) AND (CONTCOHB = -4).
COMPUTE CONTCOHB = ((M3100Y1-1980)*12) + M3100M1.
COMPUTE CHBMNTH1 = M3100M1.
COMPUTE CHBYEAR1 = M3100Y1.
END IF.

```

*<If married when started cohabiting (M45001=1), use dates for next spell of cohabitation in which respondent was not married when started to cohabit (i.e., M45002=0).>

```

DO IF (M45001=1) AND (M45002=0) AND (M3100Y1 GT 0) AND (M3100M1 GT 0).
COMPUTE CONTCOHB = ((M3100Y2-1980)*12) + M3100M2.
COMPUTE CHBMNTH1 = M3100M2.
COMPUTE CHBYEAR1 = M3100Y2.
END IF.

```

*<If respondent was not married when started to cohabit in the first loop, and had a second spell of cohabitation that did not start as a marriage, calculate the dates of the second cohabitation.>

```

DO IF (M3100Y2 GT 0) AND (M3100M2 GT 0) AND (M45001=0) AND (M45002=0).
COMPUTE CHBMNTH2 = M3100M2.

```

COMPUTE CHBYEAR2 = M3100Y2.
END IF.

*<Code all don't know/refused responses as appropriate>

```
IF (M3100M1=-1) AND (M45001=0) AND (CHBMNTH1 = -4) CHBMNTH1=-1.  
IF (M3100Y1=-1) AND (M45001=0) AND (CHBYEAR1 = -4) CHBYEAR1=-1.  
IF (M3100M1=-2) AND (M45001=0) AND (CHBMNTH1 = -4) CHBMNTH1=-2.  
IF (M3100Y1=-2) AND (M45001=0) AND (CHBYEAR1 = -4) CHBYEAR1=-2.  
IF (M3100M1=-3) AND (M45001=0) AND (CHBMNTH1 = -4) CHBMNTH1=-3.  
IF (M3100Y1=-3) AND (M45001=0) AND (CHBYEAR1 = -4) CHBYEAR1=-3.  
IF (M3100M2=-1) AND (M45001=1) AND (CHBMNTH1 = -4) CHBMNTH1=-1.  
IF (M3100Y2=-1) AND (M45001=1) AND (CHBYEAR1 = -4) CHBYEAR1=-1.  
IF (M3100M2=-2) AND (M45001=1) AND (CHBMNTH1 = -4) CHBMNTH1=-2.  
IF (M3100Y2=-2) AND (M45001=1) AND (CHBYEAR1 = -4) CHBYEAR1=-2.  
IF (M3100M2=-3) AND (M45001=1) AND (CHBMNTH1 = -4) CHBMNTH1=-3.  
IF (M3100Y2=-3) AND (M45001=1) AND (CHBYEAR1 = -4) CHBYEAR1=-3.  
IF (M3100M2=-1) AND (M45001=0) AND (M45002=0) AND (CHBMNTH1 = -4) CHBMNTH1=-1.  
IF (M3100Y2=-1) AND (M45001=0) AND (M45002=0) AND (CHBYEAR1 = -4) CHBYEAR1=-1.  
IF (M3100M2=-2) AND (M45001=0) AND (M45002=0) AND (CHBMNTH1 = -4) CHBMNTH1=-2.  
IF (M3100Y2=-2) AND (M45001=0) AND (M45002=0) AND (CHBYEAR1 = -4) CHBYEAR1=-2.  
IF (M3100M2=-3) AND (M45001=0) AND (M45002=0) AND (CHBMNTH1 = -4) CHBMNTH1=-3.  
IF (M3100Y2=-3) AND (M45001=0) AND (M45002=0) AND (CHBYEAR1 = -4) CHBYEAR1=-3.
```

*Compute the total number of cohabitations for a respondent. Initialize to zero.

```
COMPUTE TOTCOHAB = 0.  
COUNT TOTCOHAB = CHBYEAR1 CHBYEAR2 (-2, -1, 1 THRU HIGHEST).  
IF (M700=1) TOTCOHAB=-4.  
IF (M700=-3) CHBMNTH1=-3.  
IF (M700=-3) CHBMNTH2=-3.  
IF (M700=-3) CHBYEAR1=-3.  
IF (M700=-3) CHBYEAR2=-3.  
IF (M700=-3) TOTCOHAB=-3.  
IF (M700=-3) CONTCOHB=-3.  
IF (M900=1) AND (M1200<0) TOTMARRY=0.  
IF (M900=1) AND (M1200<0) TOTCOHAB=0.
```

EXECUTE.

BIOLOGICAL MOTHER'S AGE AT FIRST BIRTH/YOUTH'S BIRTH

Variables Created: CV_BIO_MOM_AGE_CHILD1
CV_BIO_MOM_AGE_YOUTH

Variables Used

This program uses the same variables as the program which creates "Youth's Relationship to Household Parent Figure(s)." These variables are listed earlier in this section.

This program creates two variables which identify the age of the youth's biological mother when she gave birth to her first child and when the youth was born.

/* First establish the relationship between youth and biomom. These are people that say they live with a bio mom - need to figure out what to do when R believe bio mom in hh but not */

```
agemomfy=-3;
smomage=-3;
momage=-3;
momid=-3;

array rela1 (i) rel1_1-rel1_19;
array rela2 (i) rel2_1-rel2_19;
array rela3 (i) rel3_1-rel3_19;
array rela4 (i) rel4_1-rel4_19;
array rela5 (i) rel5_1-rel5_19;
array rela6 (i) rel6_1-rel6_19;
array rela7 (i) rel7_1-rel7_19;
array rela8 (i) rel8_1-rel8_19;
array rela9 (i) rel9_1-rel9_19;
array rela10 (i) rel10_1-rel10_19;
array rela11 (i) rel11_1-rel11_19;
array rela12 (i) rel12_1-rel12_19;
array rela13 (i) rel13_1-rel13_19;
array rela14 (i) rel14_1-rel14_19;
array rela15 (i) rel15_1-rel15_19;
array rela16 (i) rel16_1-rel16_19;
array rela17 (i) rel17_1-rel17_19;
array rela18 (i) rel18_1-rel18_19;
array rela19 (i) rel19_1-rel19_19;
array parid (i) parid1-parid19;

/* put in corrections from parent */
do i=1 to 19;
  if yid=1 then do;
    if rela1=49 or rela1=50 then do; parid=1; end;
    end;
  if yid=2 then do;
    if rela2=49 or rela2=50 then do; parid=1; end;
    end;
  if yid=3 then do;
    if rela3=49 or rela3=50 then do; parid=1; end;
    end;
  if yid=4 then do;
    if rela4=49 or rela4=50 then do; parid=1; end;
    end;
  if yid=5 then do;
    if rela5=49 or rela5=50 then do; parid=1; end;
```

Appendix 3: Family Background Variable Creation

```
end;
end;

*correct youth's age;
if yid=1 then do;
  if birthm>scrm or (birthm=scrm and birthd>scrd) then do; age1=scry-birthy-1; end;
  if (birthm<=scrm or (birthm=scrm and birthd<=scrd)) then do; age1=scry-birthy; end;
end;
if yid=2 then do;
  if birthm>scrm or (birthm=scrm and birthd>scrd) then do; age2=scry-birthy-1; end;
  if (birthm<=scrm or (birthm=scrm and birthd<=scrd)) then do; age2=scry-birthy; end;
end;
if yid=3 then do;
  if birthm>scrm or (birthm=scrm and birthd>scrd) then do; age3=scry-birthy-1; end;
  if (birthm<=scrm or (birthm=scrm and birthd<=scrd)) then do; age3=scry-birthy; end;
end;
if yid=4 then do;
  if birthm>scrm or (birthm=scrm and birthd>scrd) then do; age4=scry-birthy-1; end;
  if (birthm<=scrm or (birthm=scrm and birthd<=scrd)) then do; age4=scry-birthy; end;
end;
if yid=5 then do;
  if birthm>scrm or (birthm=scrm and birthd>scrd) then do; age5=scry-birthy-1; end;
  if (birthm<=scrm or (birthm=scrm and birthd<=scrd)) then do; age5=scry-birthy; end;
end;

do i=1 to 19;
  if yid=1 then do;
    if parid19=1 and rel19_1=3 and gender19=2 then momid=19;
    if parid18=1 and rel18_1=3 and gender18=2 then momid=18;
    if parid17=1 and rel17_1=3 and gender17=2 then momid=17;
    if parid16=1 and rel16_1=3 and gender16=2 then momid=16;
    if parid15=1 and rel15_1=3 and gender15=2 then momid=15;
    if parid14=1 and rel14_1=3 and gender14=2 then momid=14;
    if parid13=1 and rel13_1=3 and gender13=2 then momid=13;
    if parid12=1 and rel12_1=3 and gender12=2 then momid=12;
    if parid11=1 and rel11_1=3 and gender11=2 then momid=11;
    if parid10=1 and rel10_1=3 and gender10=2 then momid=10;
    if parid9=1 and rel9_1=3 and gender9=2 then momid=9;
    if parid8=1 and rel8_1=3 and gender8=2 then momid=8;
    if parid7=1 and rel7_1=3 and gender7=2 then momid=7;
    if parid6=1 and rel6_1=3 and gender6=2 then momid=6;
    if parid5=1 and rel5_1=3 and gender5=2 then momid=5;
    if parid4=1 and rel4_1=3 and gender4=2 then momid=4;
    if parid3=1 and rel3_1=3 and gender3=2 then momid=3;
    if parid2=1 and rel2_1=3 and gender2=2 then momid=2;
    if parid1=1 and rel1_1=3 and gender1=2 then momid=1;
  end;
  if yid=2 then do;
    if parid19=1 and rel19_2=3 and gender19=2 then momid=19;
    /* and so on through parid1=1, rel1_2=3, gender1=2, and momid=1 */
  end;
  if yid=3 then do;
    if parid19=1 and rel19_3=3 and gender19=2 then momid=19;
    /* and so on through parid1=1, rel1_3=3, gender1=2, and momid=1 */
  end;
  if yid=4 then do;
    if parid19=1 and rel19_4=3 and gender19=2 then momid=19;
```

```

/* and so on through parid1=1, rel142=3, gender1=2, and momid=1 */
end;
if yid=5 then do;
  if parid19=1 and rel19_5=3 and gender19=2 then momid=19;
    /* and so on through parid1=1, rel1_5=3, gender1=2, and momid=1 */
  end;
end;

do i=1 to 19;
  if momid=.. and ymomid ne . then do;
    if yid=1 then do;
      if ymomid19=1 and rel19_1=3 and gender19=2 then momid=19;
      if ymomid18=1 and rel18_1=3 and gender18=2 then momid=18;
      if ymomid17=1 and rel17_1=3 and gender17=2 then momid=17;
      if ymomid16=1 and rel16_1=3 and gender16=2 then momid=16;
      if ymomid15=1 and rel15_1=3 and gender15=2 then momid=15;
      if ymomid14=1 and rel14_1=3 and gender14=2 then momid=14;
      if ymomid13=1 and rel13_1=3 and gender13=2 then momid=13;
      if ymomid12=1 and rel12_1=3 and gender12=2 then momid=12;
      if ymomid11=1 and rel11_1=3 and gender11=2 then momid=11;
      if ymomid10=1 and rel10_1=3 and gender10=2 then momid=10;
      if ymomid9=1 and rel9_1=3 and gender9=2 then momid=9;
      if ymomid8=1 and rel8_1=3 and gender8=2 then momid=8;
      if ymomid7=1 and rel7_1=3 and gender7=2 then momid=7;
      if ymomid6=1 and rel6_1=3 and gender6=2 then momid=6;
      if ymomid5=1 and rel5_1=3 and gender5=2 then momid=5;
      if ymomid4=1 and rel4_1=3 and gender4=2 then momid=4;
      if ymomid3=1 and rel3_1=3 and gender3=2 then momid=3;
      if ymomid2=1 and rel2_1=3 and gender2=2 then momid=2;
      if ymomid1=1 and rel1_1=3 and gender1=2 then momid=1;
    end;
    if yid=2 then do;
      if ymomid19=1 and rel19_2=3 and gender19=2 then momid=19;
        /* and so on through ymomid1=1, rel1_2=3, gender1=2, and momid=1 */
    end;
  end;
  if yid=3 then do;
    if ymomid19=1 and rel19_3=3 and gender19=2 then momid=19;
    /* and so on through ymomid1=1, rel1_3=3, gender1=2, and momid=1 */
  end;
  if yid=4 then do;
    if ymomid19=1 and rel19_4=3 and gender19=2 then momid=19;
    /* and so on through ymomid1=1, rel1_4=3, gender1=2, and momid=1 */
  end;
  if yid=5 then do;
    if ymomid19=1 and rel19_5=3 and gender19=2 then momid=19;
    /* and so on through ymomid1=1, rel1_5=3, gender1=2, and momid=1 */
  end;
end;

/* Connect the age of the mom to the correct id */
/*define for nonresident mom */

if ynrmomid=9 and ngend9=2 then momage=nage9;
if ynrmomid=8 and ngend8=2 then momage=nage8;
if ynrmomid=7 and ngend7=2 then momage=nage7;
if ynrmomid=6 and ngend6=2 then momage=nage6;

```

Appendix 3: Family Background Variable Creation

```
if ynrmomid=5 and ngend5=2 then momage=nage5;
if ynrmomid=4 and ngend4=2 then momage=nage4;
if ynrmomid=3 and ngend3=2 then momage=nage3;
if ynrmomid=2 and ngend2=2 then momage=nage2;
if ynrmomid=1 and ngend1=2 then momage=nage1;

if pnrmomid=9 and ngend9=2 then momage=nage9;
if pnrmomid=8 and ngend8=2 then momage=nage8;
if pnrmomid=7 and ngend7=2 then momage=nage7;
if pnrmomid=6 and ngend6=2 then momage=nage6;
if pnrmomid=5 and ngend5=2 then momage=nage5;
if pnrmomid=4 and ngend4=2 then momage=nage4;
if pnrmomid=3 and ngend3=2 then momage=nage3;
if pnrmomid=2 and ngend2=2 then momage=nage2;
if pnrmomid=1 and ngend1=2 then momage=nage1;

/* define for mom in household */

if momid=1 and doby1 ne -3 then momage=age1;
if momid=2 and doby2 ne -3 then momage=age2;
if momid=3 and doby3 ne -3 then momage=age3;
if momid=4 and doby4 ne -3 then momage=age4;
if momid=5 and doby5 ne -3 then momage=age5;
if momid=6 and doby6 ne -3 then momage=age6;
if momid=7 and doby7 ne -3 then momage=age7;
if momid=8 and doby8 ne -3 then momage=age8;
if momid=9 and doby9 ne -3 then momage=age9;
if momid=10 and doby10 ne -3 then momage=age10;
if momid=11 and doby11 ne -3 then momage=age11;
if momid=12 and doby12 ne -3 then momage=age12;
if momid=13 and doby13 ne -3 then momage=age13;
if momid=14 and doby14 ne -3 then momage=age14;
if momid=15 and doby15 ne -3 then momage=age15;
if momid=16 and doby16 ne -3 then momage=age16;
if momid=17 and doby17 ne -3 then momage=age17;
if momid=18 and doby18 ne -3 then momage=age18;
if momid=19 and doby19 ne -3 then momage=age19;
if momid=20 and doby20 ne -3 then momage=age20;

if pi025=0 and pi058y>0 or pi058m>0 then do;
  if momid=pi020 or (momid=parentid and pi020=-4) then do;
    if pi058y>0 and scry>1996 and (pi058m>scrm or (pi058m=scrm and pi058d>scrd)) then do;
      momage=scry-pi058y-1; agefl=1; end;
    if pi058y>0 and scry>1996 and (pi058m<=scrm or (pi058m=scrm and pi058d<=scrd)) then do;
      momage=scry-pi058y; agefl=1; end;
  end;
end;

/* loop through looking for brothers and sisters (13, 14, 15, 18), if we have these, and there are no negative value
relationships, and we have a valid year of birth on everyone, then we can find the oldest's age and subtract from
mom's age*/;

array age (i) age1-age19; array sage (i) sage1-sage19;

do i=1 to 19;
  sage=-4;
  if yid=1 and momage>0 then do;
```

```

if (rela1=13 or rela1=14 or rela1=15 or rela1=18) and age ge 0 then sage=momage-age;
if rela1=20 or rela1=17 then sage=-3;
if (rela1=13 or rela1=14 or rela1=15 or rela1=18) and age<0 then sage=-4;
end;
if yid=2 and momage>0 then do;
if (rela2=13 or rela2=14 or rela2=15 or rela2=18) and age ge 0 then sage=momage-age;
if rela2=20 or rela2=17 then sage=-3;
if (rela2=13 or rela2=14 or rela2=15 or rela2=18) and age<0 then sage=-4;
end;
if yid=3 and momage>0 then do;
if (rela3=13 or rela3=14 or rela3=15 or rela3=18) and age ge 0 then sage=momage-age;
if rela3=20 or rela3=17 then sage=-3;
if (rela3=13 or rela3=14 or rela3=15 or rela3=18) and age<0 then sage=-4;
end;
if yid=4 and momage>0 then do;
if (rela4=13 or rela4=14 or rela4=15 or rela4=18) and age ge 0 then sage=momage-age;
if rela4=20 or rela4=17 then sage=-3;
if (rela4=13 or rela4=14 or rela4=15 or rela4=18) and age<0 then sage=-4;
end;
if yid=5 and momage>0 then do;
if (rela5=13 or rela5=14 or rela5=15 or rela5=18) and age ge 0 then sage=momage-age;
if rela5=20 or rela5=17 then sage=-3;
if (rela5=13 or rela5=14 or rela5=15 or rela5=18) and age<0 then sage=-4;
end;
end;

array nrel (i) nrel1-nrel23; array nage (i) nage1-nage23; array nrsage (i) nrsage1-nrsage23;

do i=1 to 23;
nrsage=-4;
if momage>0 then do;
if (nrel=13 or nrel=14 or nrel=15 or nrel=18) and nage ge 0 then nrsage=momage-nage;
if nrel=20 or nrel=17 then nrsage=-3;
if (nrel=13 or nrel=14 or nrel=15 or nrel=18) and nage<0 then nrsage=-3;
end;
end;

/* To get youths birth: if mom died, find yeardif by subtracting the year of death from the Rs birthyear, then subtract
this difference from the age at the time of death to get to the bio mom's age at Rs birth */

if yid=1 and momage>0 and age1>0 then do; sage1=momage-age1; end;
if yid=2 and momage>0 and age2>0 then do; sage2=momage-age2; end;
if yid=3 and momage>0 and age3>0 then do; sage3=momage-age3; end;
if yid=4 and momage>0 and age4>0 then do; sage4=momage-age4; end;
if yid=5 and momage>0 and age5>0 then do; sage5=momage-age5; end;

unadj1=sage1;
unadj2=sage2;
unadj3=sage3;
unadj4=sage4;
unadj5=sage5;

array sn76 (i) sn761-sn763; array sn79 (i) sn791-sn793;

do i=1 to 3;
if sn701=0 and yid=1 and (sn76 ge birthy) then do;
yeardif=sn76-birthy; sage1=sn79-yeardif; end;

```

```

if sn702=0 and yid=2 and (sn76 ge birthy) then do;
  yeardif=sn76-birthy; sage2=sn79-yeardif; end;
if sn703=0 and yid=3 and (sn76 ge birthy) then do;
  yeardif=sn76-birthy; sage3=sn79-yeardif; end;
if sn704=0 and yid=4 and (sn76 ge birthy) then do;
  yeardif=sn76-birthy; sage4=sn79-yeardif; end;
if sn705=0 and yid=5 and (sn76 ge birthy) then do;
  yeardif=sn76-birthy; sage5=sn79-yeardif; end;
end;

array sdif (i) sdif1-sdif23;

do i=1 to 23;
  if sn701=0 and yid=1 and yeardif>0 and yeardif ne . and nrsage>0 then do;
    sdif=unadj1-nrsage; nrsage=sage1-sdif; end;
  if sn702=0 and yid=2 and yeardif>0 and yeardif ne . and nrsage>0 then do;
    sdif=unadj2-nrsage; nrsage=sage2-sdif; end;
  if sn703=0 and yid=3 and yeardif>0 and yeardif ne . and nrsage>0 then do;
    sdif=unadj3-nrsage; nrsage=sage3-sdif; end;
  if sn704=0 and yid=4 and yeardif>0 and yeardif ne . and nrsage>0 then do;
    sdif=unadj4-nrsage; nrsage=sage4-sdif; end;
  if sn705=0 and yid=5 and yeardif>0 and yeardif ne . and nrsage>0 then do;
    sdif=unadj5-nrsage; nrsage=sage5-sdif; end;
end;

array sibdif (i) sibdif1-sibdif23;
do i=1 to 19;
  sibdif=0;
end;

do i=2 to 19;
  if sn701=0 and yid=1 and yeardif>0 and yeardif ne . and sage>0 then do;
    sibdif=unadj1-sage; sage=sage1-sibdif; end;
end;
do i=3 to 19;
  if sn702=0 and yid=2 and yeardif>0 and yeardif ne . and sage>0 then do;
    sibdif=unadj2-sage; sibdif1=unadj2-unadj1; sage=sage2-sibdif; sage1=sage2-sibdif1; end;
end;
do i=4 to 19;
  if sn703=0 and yid=3 and yeardif>0 and yeardif ne . and sage>0 then do;
    sibdif=unadj3-sage; sibdif1=unadj3-unadj1; sibdif2=unadj3-unadj2;
    sage=sage3-sibdif; sage1=sage3-sibdif1; sage2=sage3-sibdif2;
  end;
end;
do i=5 to 19;
  if sn704=0 and yid=4 and yeardif>0 and yeardif ne . and sage>0 then do;
    sibdif=unadj4-sage; sibdif1=unadj4-unadj1; sibdif2=unadj4-unadj2; sibdif3=unadj4-unadj3;
    sage=sage4-sibdif; sage1=sage4-sibdif1; sage2=sage4-sibdif2; sage3=sage4-sibdif3;
  end;
end;
do i=6 to 19;
  if sn705=0 and yid=5 and yeardif>0 and yeardif ne . and sage>0 then do;
    sibdif=unadj5-sage; sibdif1=unadj5-unadj1; sibdif2=unadj5-unadj2; sibdif3=unadj5-unadj3;
    sibdif4=unadj5-unadj4;
    sage=sage5-sibdif; sage1=sage5-sibdif1; sage2=sage5-sibdif2; sage3=sage5-sibdif3; sage4=sage5-sibdif4;
  end;
end;

```

```

/* Recalculate Youth variable and age */
if yid=1 and sage1>0 then do; smomage=sage1; end;
if yid=2 and sage2>0 then do; smomage=sage2; end;
if yid=3 and sage3>0 then do; smomage=sage3; end;
if yid=4 and sage4>0 then do; smomage=sage4; end;
if yid=5 and sage5>0 then do; smomage=sage5; end;

do i=1 to 19;
    if sage=-4 then do; sage=999; end;
    if (sage<0 and sage ne -4) or sage=0 then do; sage=0; end;
end;
do i=1 to 23;
    if nrsage=-4 then do; nrsage=999; end;
    if (nrsage<0 and nrsage ne -4) or nrsage=0 then do; nrsage=0; end;
end;

if sage1>0 and sage2>0 and sage3>0 and sage4>0 and sage5>0 and sage6>0 and sage7>0 and sage8>0 and sage9>0
    and sage10>0 and sage11>0 and sage12>0 and sage13>0 and sage14>0 and sage15>0 and sage16>0 and
    sage17>0 and sage18>0 and sage19>0 and nrsage1>0 and nrsage2>0 and nrsage3>0 and nrsage4>0 and
    nrsage5>0 and nrsage6>0 and nrsage7>0 and nrsage8>0 and nrsage9>0 and nrsage10>0 and nrsage11>0
    and nrsage12>0 and nrsage13>0 and nrsage14>0 and nrsage15>0 and nrsage16>0 and nrsage17>0 and
    nrsage18>0 and nrsage19>0 and nrsage20>0 and nrsage21>0 and nrsage22>0 and nrsage23>0 then do;
    agemomfy=min(sage1, sage2, sage3, sage4, sage5, sage6, sage7, sage8, sage9, sage10, sage11, sage12, sage13,
        sage14, sage15, sage16, sage17, sage18, sage19, nrsage1, nrsage2, nrsage3, nrsage4, nrsage5, nrsage6,
        nrsage7, nrsage8, nrsage9, nrsage10, nrsage11, nrsage12, nrsage13, nrsage14, nrsage15, nrsage16, nrsage17,
        nrsage18, nrsage19, nrsage20, nrsage21, nrsage22, nrsage23);
end;

if agemomfy=999 then agemomfy=-3;

/*youth says doesn't live with bio mom and no survey information collected on non-res bio mom*/
if sn701=1 and yid=1 then do;
    if (saq13=1 and saq14 ne 1) or (saq13<1 and saq13>-4) then do;
        smomage=-3; agemomfy=-3; end;
end;
if sn702=1 and yid=2 then do;
    if (saq13=1 and saq14 ne 1) or (saq13<1 and saq13>-4) then do;
        smomage=-3; agemomfy=-3; end;
end;
if sn703=1 and yid=3 then do;
    if (saq13=1 and saq14 ne 1) or (saq13<1 and saq13>-4) then do;
        smomage=-3; agemomfy=-3; end;
end;
if sn704=1 and yid=4 then do;
    if (saq13=1 and saq14 ne 1) or (saq13<1 and saq13>-4) then do;
        smomage=-3; agemomfy=-3; end;
end;
if sn705=5 and yid=5 then do;
    if (saq13=1 and saq14 ne 1) or (saq13<1 and saq13>-4) then do;
        smomage=-3; agemomfy=-3; end;
endsas;

```

YOUTH CITIZENSHIP STATUS

Variables Created: CV_CITIZENSHIP

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
pubid	PUBID	dob01y	KEY!BDATE_Y
pinf015y	PINF-015_Y	spparid1-6	HHI2_SPOPARID.01-.16
pinf020	PINF-020	hh1rel1-5	HHI2_REL1.01-.05
pinf096	PINF-096	hh2rel1-5	HHI2_REL2.01-.05
pinf097	PINF-097	hh3rel1-5	HHI2_REL3.01-.05
pinf159	PINF-159	hh4rel1-5	HHI2_REL4.01-.05
pinf160	PINF-160	hh5rel1-5	HHI2_REL5.01-.05
y01hhid	PARYOUTH_HHID	hh6rel1-5	HHI2_REL6.01-.05
ythpar01	PARYOUTH_PARENT	hh7rel1-5	HHI2_REL7.01-.05
ythparid	PARYOUTH_PARENTID	hh8rel1-5	HHI2_REL8.01-.05
ythprsex	PARYOUTH_PARENTSEX	hh9rel1-5	HHI2_REL9.01-.05
p2001	P2-001	hh10rel1-5	HHI2_REL10.01-.05
p2002	P2-002	hh11rel1-5	HHI2_REL11.01-.05
p2008	P2-008	hh12rel1-5	HHI2_REL12.01-.05
p2037	P2-037	hh13rel1-5	HHI2_REL13.01-.05
p2038	P2-038	hh14rel1-5	HHI2_REL14.01-.05
p2044	P2-044	hh15rel1-5	HHI2_REL15.01-.05
p2108b01	P2-108B.01	hh16rel1-5	HHI2_REL16.01-.05

Codes for Created Variable

1 = Citizen, born in the U.S.

2 = Unknown, not born in the U.S.

3 = Unknown, can't determine birthplace

This program uses data from the parent interview to determine the youth's U.S. citizenship status. If the responding parent is the youth's biological parent, the responding parent's citizenship status is used to ascertain the youth's citizenship status. If the responding parent is not a bio parent, the nonresponding bio parent's citizenship status is used to determine the youth's status when available. If the responding parent is not a bio parent and nonresponding bio parent information is not available, the youth's citizenship is coded as "can't determine."

**** PARENT IDENTIFICATION SECTION ****

*<Initialize everyone to zero>

Compute parelate=0.

*<ythparid=1>

```
do if (ythparid=1 and pinf020<0) or (pinf020=1).
if (y01hhid=2) and (spparid1=3) parelate=hh3rel2.
if (y01hhid=2) and (spparid1=4) parelate=hh4rel2.
if (y01hhid=2) and (spparid1=5) parelate=hh5rel2.
if (y01hhid=2) and (spparid1=6) parelate=hh6rel2.
if (y01hhid=2) and (spparid1=7) parelate=hh7rel2.
if (y01hhid=2) and (spparid1=8) parelate=hh8rel2.
if (y01hhid=2) and (spparid1=9) parelate=hh9rel2.
if (y01hhid=2) and (spparid1=10) parelate=hh10rel2.
if (y01hhid=2) and (spparid1=11) parelate=hh11rel2.
if (y01hhid=2) and (spparid1=12) parelate=hh12rel2.
if (y01hhid=2) and (spparid1=13) parelate=hh13rel2.
if (y01hhid=2) and (spparid1=14) parelate=hh14rel2.
if (y01hhid=2) and (spparid1=15) parelate=hh15rel2.
```

```
if (y01hhid=2) and (spparid1=16) parelate=hh16rel2.
if (y01hhid=3) and (spparid1=2) parelate=hh2rel3.
if (y01hhid=3) and (spparid1=4) parelate=hh4rel3.
if (y01hhid=3) and (spparid1=5) parelate=hh5rel3.
if (y01hhid=3) and (spparid1=6) parelate=hh6rel3.
if (y01hhid=3) and (spparid1=7) parelate=hh7rel3.
if (y01hhid=3) and (spparid1=8) parelate=hh8rel3.
if (y01hhid=3) and (spparid1=9) parelate=hh9rel3.
if (y01hhid=3) and (spparid1=10) parelate=hh10rel3.
if (y01hhid=3) and (spparid1=11) parelate=hh11rel3.
if (y01hhid=3) and (spparid1=12) parelate=hh12rel3.
if (y01hhid=3) and (spparid1=13) parelate=hh13rel3.
if (y01hhid=3) and (spparid1=14) parelate=hh14rel3.
if (y01hhid=3) and (spparid1=15) parelate=hh15rel3.
if (y01hhid=3) and (spparid1=16) parelate=hh16rel3.
if (y01hhid=4) and (spparid1=2) parelate=hh2rel4.
if (y01hhid=4) and (spparid1=3) parelate=hh3rel4.
if (y01hhid=4) and (spparid1=5) parelate=hh5rel4.
if (y01hhid=4) and (spparid1=6) parelate=hh6rel4.
if (y01hhid=4) and (spparid1=7) parelate=hh7rel4.
```

```

if (y01hhid=4) and (spparid1=8) parelate=hh8rel4.
if (y01hhid=4) and (spparid1=9) parelate=hh9rel4.
if (y01hhid=4) and (spparid1=10) parelate=hh10rel4.
if (y01hhid=4) and (spparid1=11) parelate=hh11rel4.
if (y01hhid=4) and (spparid1=12) parelate=hh12rel4.
if (y01hhid=4) and (spparid1=13) parelate=hh13rel4.
if (y01hhid=4) and (spparid1=14) parelate=hh14rel4.
if (y01hhid=4) and (spparid1=15) parelate=hh15rel4.
if (y01hhid=4) and (spparid1=16) parelate=hh16rel4.
if (y01hhid=5) and (spparid1=2) parelate=hh2rel5.
if (y01hhid=5) and (spparid1=3) parelate=hh3rel5.
if (y01hhid=5) and (spparid1=4) parelate=hh4rel5.
if (y01hhid=5) and (spparid1=6) parelate=hh6rel5.
if (y01hhid=5) and (spparid1=7) parelate=hh7rel5.
if (y01hhid=5) and (spparid1=8) parelate=hh8rel5.
if (y01hhid=5) and (spparid1=9) parelate=hh9rel5.
if (y01hhid=5) and (spparid1=10) parelate=hh10rel5.
if (y01hhid=5) and (spparid1=11) parelate=hh11rel5.
if (y01hhid=5) and (spparid1=12) parelate=hh12rel5.
if (y01hhid=5) and (spparid1=13) parelate=hh13rel5.
if (y01hhid=5) and (spparid1=14) parelate=hh14rel5.
if (y01hhid=5) and (spparid1=15) parelate=hh15rel5.
if (y01hhid=5) and (spparid1=16) parelate=hh16rel5.

```

*< Due to space restrictions, the remaining relationship loops are not reproduced in their entirety. The pattern for each loop is similar to that seen above in ythparid=1. Users who need the complete code for this variable should contact NLS User Services.>

```

*<ythparid=2>
else if (ythparid=2 and pinf020<0) or (pinf020=2).
if (y01hhid=1) and (spparid2=3) parelate=hh3rel1.
    *< and so on through spparid2=16 and hh16rel1 >
if (y01hhid=3) and (spparid2=1) parelate=hh1rel3.
    *< and so on through spparid2=16 and hh16rel3 >
if (y01hhid=4) and (spparid2=1) parelate=hh1rel4.
    *< and so on through spparid2=16 and hh16rel4 >
if (y01hhid=5) and (spparid2=1) parelate=hh1rel5.
    *< and so on through hh16rel5 >

```

```

*<ythparid=3>
else if (ythparid=3 and pinf020<0) or (pinf020=3).
if (y01hhid=1) and (spparid3=2) parelate=hh2rel1.
    *< and so on through spparid3=16 and hh16rel1 >
if (y01hhid=2) and (spparid3=1) parelate=hh1rel2.
    *< and so on through spparid3=16 and hh16rel2 >
if (y01hhid=4) and (spparid3=1) parelate=hh1rel4.
    *< and so on through spparid3=16 and hh16rel4 >
if (y01hhid=5) and (spparid3=1) parelate=hh1rel5.
    *< and so on through spparid3=16 and hh16rel5 >

```

```

*<ythparid=4>
else if (ythparid=4 and pinf020<0) or (pinf020=4).
if (y01hhid=1) and (spparid4=2) parelate=hh2rel1.
    *< and so on through spparid4=16 and hh16rel1 >
if (y01hhid=2) and (spparid4=1) parelate=hh1rel2.

```

```

    *< and so on through spparid4=16 and hh16rel2 >
if (y01hhid=3) and (spparid4=1) parelate=hh1rel3.
    *< and so on through spparid4=16 and hh16rel3 >
if (y01hhid=5) and (spparid4=1) parelate=hh1rel5.
    *< and so on through spparid4=16 and hh16rel5 >

```

```

*<ythparid=5>
else if ((ythparid=5 and pinf020<0) or (pinf020=5)).
if (y01hhid=1) and (spparid5=2) parelate=hh2rel1.
    *< and so on through spparid5=16 and hh16rel1 >
if (y01hhid=2) and (spparid5=1) parelate=hh1rel2.
    *< and so on through spparid5=16 and hh16rel2 >
if (y01hhid=3) and (spparid5=1) parelate=hh1rel3.
    *< and so on through spparid5=16 and hh16rel3 >
if (y01hhid=4) and (spparid5=1) parelate=hh1rel4.
    *< and so on through spparid5=16 and hh16rel4 >

```

```

*<ythparid=6>
else if ((ythparid=6 and pinf020<0) or (pinf020=6)).
if (y01hhid=1) and (spparid6=2) parelate=hh2rel1.
    *< and so on through spparid6=16 and hh16rel1 >
if (y01hhid=2) and (spparid6=1) parelate=hh1rel2.
    *< and so on through spparid6=16 and hh16rel2 >
if (y01hhid=3) and (spparid6=1) parelate=hh1rel3.
    *< and so on through spparid6=16 and hh16rel3 >
if (y01hhid=4) and (spparid6=1) parelate=hh1rel4.
    *< and so on through spparid6=16 and hh16rel4 >
if (y01hhid=5) and (spparid6=1) parelate=hh1rel5.
    *< and so on through spparid6=16 and hh16rel5 >

```

```

*<ythparid=7>
else if ((ythparid=7 and pinf020<0) or (pinf020=7)).
if (y01hhid=1) and (spparid7=2) parelate=hh2rel1.
    *< and so on through spparid7=16 and hh16rel1 >
if (y01hhid=2) and (spparid7=1) parelate=hh1rel2.
    *< and so on through spparid7=16 and hh16rel2 >
if (y01hhid=3) and (spparid7=1) parelate=hh1rel3.
    *< and so on through spparid7=16 and hh16rel3 >
if (y01hhid=4) and (spparid7=1) parelate=hh1rel4.
    *< and so on through spparid7=16 and hh16rel4 >
if (y01hhid=5) and (spparid7=1) parelate=hh1rel5.
    *< and so on through spparid7=16 and hh16rel5 >

```

```

*<ythparid=8>
else if ((ythparid=8 and pinf020<0) or (pinf020=8)).
if (y01hhid=1) and (spparid8=2) parelate=hh2rel1.
    *< and so on through spparid8=16 and hh16rel1 >
if (y01hhid=2) and (spparid8=1) parelate=hh1rel2.
    *< and so on through spparid8=16 and hh16rel2 >
if (y01hhid=4) and (spparid8=1) parelate=hh1rel4.
    *< and so on through spparid8=16 and hh16rel4 >
if (y01hhid=5) and (spparid8=1) parelate=hh1rel5.
    *< and so on through spparid8=16 and hh16rel5 >

```

```

*<ythparid=9>
else if ((ythparid=9 and pinf020<0) or (pinf020=9)).

```

```

if (y01hhid=1) and (spparid9=2) parelate=hh2rel1.
  *< and so on through spparid9=16 and hh16rel1 >
if (y01hhid=2) and (spparid9=1) parelate=hh1rel2.
  *< and so on through spparid9=16 and hh16rel2 >
if (y01hhid=3) and (spparid9=1) parelate=hh1rel3.
  *< and so on through spparid9=16 and hh16rel3 >
if (y01hhid=4) and (spparid9=1) parelate=hh1rel4.
  *< and so on through spparid9=16 and hh16rel4 >
if (y01hhid=5) and (spparid9=1) parelate=hh1rel5.
  *< and so on through spparid9=16 and hh16rel5 >

```

******* PARENT STATUS SECTION *******

```

*< Determine legal status of partner/spouse of
responding parent>
```

```

*< initialize everyone to valid skip>
compute slegmom = -4.
compute slegdad = -4.
```

```

*< use data from parelate to determine legal status of
spouse/partner of rp>
if (parelate=-1 and ythprsex=2) slegdad=-1.
if (parelate=-2 and ythprsex=2) slegdad=-2.
if (parelate=-1 and ythprsex=1) slegmom=-1.
if (parelate=-2 and ythprsex=1) slegmom=-2.
if (parelate=3) slegdad=1.
if (parelate=4) slegmom=1.
if (parelate=5) slegdad=3.
if (parelate=6) slegmom=3.
if (parelate=7) slegdad=2.
if (parelate=8) slegmom=2.
if (parelate=9) slegdad=4.
if (parelate=10) slegmom=4.
if (parelate>10 and ythprsex=1) slegdad=4.
if (parelate>10 and ythprsex=2) slegmom=4.
```

```

*< Determine legal status of responding parent>
```

```

*<initialize everyone to valid skip>
compute legmom = -4.
compute legdad = -4.
```

```

*<include loops to account for corrections to the
responding parent's hhid>
```

```

*<updated parent id = 1>
do if (pinf020=1) and (y01hhid=2).
  if (hh1rel2=3) legmom=1.
  if (hh1rel2=4) legdad=1.
  if (hh1rel2=5) legmom=3.
  if (hh1rel2=6) legdad=3.
  if (hh1rel2=7) legmom=2.
  if (hh1rel2=8) legdad=2.
  if (hh1rel2=9) legmom=4.
  if (hh1rel2=10) legdad=4.
  if (hh1rel2>10) legmom=13.
```

```

else if (pinf020=1) and (y01hhid=3).
  if (hh1rel3=3) legmom=1.
  if (hh1rel3=4) legdad=1.
  if (hh1rel3=5) legmom=3.
  if (hh1rel3=6) legdad=3.
  if (hh1rel3=7) legmom=2.
  if (hh1rel3=8) legdad=2.
  if (hh1rel3=9) legmom=4.
  if (hh1rel3=10) legdad=4.
  if (hh1rel3>10 and ythprsex=2) legmom=4.
  if (hh1rel3>10 and ythprsex=1) legdad=4.
else if (pinf020=1) and (y01hhid=4).
  if (hh1rel4=3) legmom=1.
  if (hh1rel4=4) legdad=1.
  if (hh1rel4=5) legmom=3.
  if (hh1rel4=6) legdad=3.
  if (hh1rel4=7) legmom=2.
  if (hh1rel4=8) legdad=2.
  if (hh1rel4=9) legmom=4.
  if (hh1rel4=10) legdad=4.
  if (hh1rel4>10 and ythprsex=2) legmom=4.
  if (hh1rel4>10 and ythprsex=1) legdad=4.
else if (pinf020=1) and (y01hhid=5).
  if (hh1rel5=3) legmom=1.
  if (hh1rel5=4) legdad=1.
  if (hh1rel5=5) legmom=3.
  if (hh1rel5=6) legdad=3.
  if (hh1rel5=7) legmom=2.
  if (hh1rel5=8) legdad=2.
  if (hh1rel5=9) legmom=4.
  if (hh1rel5=10) legdad=4.
  if (hh1rel5>10 and ythprsex=2) legmom=4.
  if (hh1rel5>10 and ythprsex=1) legdad=4.
```

*< At this point the program loops through the same 10 statements for each combination of youth and parent id. These loops are truncated here due to space considerations. Users who need the entire program should contact NLS User Services. >

```

*<updated parent id = 2>
else if (pinf020=2) and (y01hhid=1).
  if (hh2rel1=3) legmom=1.
else if (pinf020=2) and (y01hhid=3).
  if (hh2rel3=3) legmom=1. *< and so on>
else if (pinf020=2) and (y01hhid=4).
  if (hh2rel4=3) legmom=1. *< and so on>
else if (pinf020=2) and (y01hhid=5).
  if (hh2rel5=3) legmom=1. *< and so on>
```

```

*<updated parent id = 3>
else if (pinf020=3) and (y01hhid=1).
  if (hh3rel1=3) legmom=1. *< and so on>
else if (pinf020=3) and (y01hhid=2).
  if (hh3rel2=3) legmom=1. *< and so on>
else if (pinf020=3) and (y01hhid=4).
  if (hh3rel4=3) legmom=1. *< and so on>
```

```

else if (pinf020=3) and (y01hhid=5).
    if (hh3rel5=3) legmom=1. *< and so on>

*<updated parent id = 4>
else if (pinf020=4) and (y01hhid=1).
    if (hh4rel1=3) legmom=1. *< and so on>
else if (pinf020=4) and (y01hhid=2).
    if (hh4rel2=3) legmom=1. *< and so on>
else if (pinf020=4) and (y01hhid=3).
    if (hh4rel3=3) legmom=1. *< and so on>
else if (pinf020=4) and (y01hhid=5).
    if (hh4rel5=3) legmom=1. *< and so on>

*<updated parent id = 5*>< and so on>
else if (pinf020=5) and (y01hhid=1).
    if (hh5rel1=3) legmom=1. *< and so on>
else if (pinf020=5) and (y01hhid=2).
    if (hh5rel2=3) legmom=1. *< and so on>
else if (pinf020=5) and (y01hhid=3).
    if (hh5rel3=3) legmom=1. *< and so on>
else if (pinf020=5) and (y01hhid=4).
    if (hh5rel4=3) legmom=1. *< and so on>

*<updated parent id = 6>
else if (pinf020=6) and (y01hhid=1).
    if (hh6rel1=3) legmom=1. *< and so on>
else if (pinf020=6) and (y01hhid=2).
    if (hh6rel2=3) legmom=1. *< and so on>
else if (pinf020=6) and (y01hhid=3).
    if (hh6rel3=3) legmom=1. *< and so on>
else if (pinf020=6) and (y01hhid=4).
    if (hh6rel4=3) legmom=1. *< and so on>
else if (pinf020=6) and (y01hhid=5).
    if (hh6rel5=3) legmom=1. *< and so on>

*<updated parent id = 7>
else if (pinf020=7) and (y01hhid=1).
    if (hh7rel1=3) legmom=1. *< and so on>
else if (pinf020=7) and (y01hhid=2).
    if (hh7rel2=3) legmom=1. *< and so on>
else if (pinf020=7) and (y01hhid=3).
    if (hh7rel3=3) legmom=1. *< and so on>
else if (pinf020=7) and (y01hhid=4).
    if (hh7rel4=3) legmom=1. *< and so on>
else if (pinf020=7) and (y01hhid=5).
    if (hh7rel5=3) legmom=1. *< and so on>

*<updated parent id = 8>
else if (pinf020=8) and (y01hhid=1).
    if (hh8rel1=3) legmom=1. *< and so on>
else if (pinf020=8) and (y01hhid=2).
    if (hh8rel2=3) legmom=1. *< and so on>
else if (pinf020=8) and (y01hhid=3).
    if (hh8rel3=3) legmom=1. *< and so on>
else if (pinf020=8) and (y01hhid=4).
    if (hh8rel4=3) legmom=1. *< and so on>
else if (pinf020=8) and (y01hhid=5).

if (hh8rel5=3) legmom=1. *< and so on>

*<updated parent id = 9>
else if (pinf020=9) and (y01hhid=1).
    if (hh9rel1=3) legmom=1. *< and so on>
else if (pinf020=9) and (y01hhid=2).
    if (hh9rel2=3) legmom=1. *< and so on>
else if (pinf020=9) and (y01hhid=3).
    if (hh9rel3=3) legmom=1. *< and so on>
else if (pinf020=9) and (y01hhid=4).
    if (hh9rel4=3) legmom=1. *< and so on>
else if (pinf020=9) and (y01hhid=5).
    if (hh9rel5=3) legmom=1. *< and so on>

*<if there is valid data for the mother, use it.>
if (ythpar01=-1 and pinf020=-4 and pinf097=-4 and
    pinf160=-4) and (ythprsex=2) legmom=-1.
if (ythpar01=-2 and pinf020=-4 and pinf097=-4 and
    pinf160=-4) and (ythprsex=2) legmom=-2.
do if any(ythpar01, 0,1,3,5,7,9,11,13,15,17,19,21) and
    (pinf097=-4 and pinf160=-4) and (legmom=-4) and
    (ythprsex=2).
    if (ythpar01=1) legmom=1.
    if (ythpar01=3) legmom=2.
    if (ythpar01=5) legmom=3.
    if (ythpar01>=7) legmom=4.
else if (pinf097>0 or pinf160>0).
    if (pinf097=1 or pinf160=1) legmom=1.
    if (pinf097=3 or pinf160=3) legmom=2.
    if (pinf097=5 or pinf160=5) legmom=3.
    if (pinf097>=7 or pinf160>=7) legmom=4.
end if.

*<use info from youth roster if paryouth info not
available>
do if any(paryth01, 0,1,3,5,7,9,11,13,15,17,19,21) and
    (pinf097=-4 and pinf160=-4) and (legmom=-4) and
    (ythpar01=-4) and (ythparid=-4).
    if (paryth01=1) legmom=1.
    if (paryth01=3) legmom=2.
    if (paryth01=5) legmom=3.
    if (paryth01>=7) legmom=4.
end if.

*<if there is valid data for the father, use it.>
if (ythpar01=-1 and pinf020=-4 and pinf096=-4 and
    pinf159=-4) and (ythprsex=1) legdad=-1.
if (ythpar01=-2 and pinf020=-4 and pinf096=-4 and
    pinf159=-4) and (ythprsex=1) legdad=-2.
do if any(ythpar01, 0,2,4,6,8,10,12,14,16,18,20,22)
    and (pinf096=-4 and pinf159=-4) and (legdad=-4)
    and (ythprsex=1).
    if (ythpar01=2) legdad=1.
    if (ythpar01=4) legdad=2.
    if (ythpar01=6) legdad=3.
    if (ythpar01>=8) legdad=4.
else if (pinf096>0 or pinf159>0).

```

Appendix 3: Family Background Variable Creation

```
if (pinf096=2 or pinf159=2) legdad=1.  
if (pinf096=4 or pinf159=4) legdad=2.  
if (pinf096=6 or pinf159=6) legdad=3.  
if (pinf096>=8 or pinf159>=8) legdad=4.  
end if.  
  
*< use info from youth roster if paryouth info not  
available >  
do if any(paryth01, 0,2,4,6,8,10,12,14,16,18,20,22)  
and (pinf096=-4 and pinf159=-4) and (legdad=-4)  
and (ythpar01=-4) and (ythparid=-4).  
    if (paryth01=2) legdad=1.  
    if (paryth01=4) legdad=2.  
    if (paryth01=6) legdad=3.  
    if (paryth01>=8) legdad=4.  
end if.  
  
*****YOUTH'S CITIZENSHIP SECTION*****  
  
*determine citizenship of youth based on bioparent  
citizenship or youth's birth in the u.s.  
  
*<initialize everyone to can't determine ythcitiz=3.>  
compute ythcitiz=-4.  
  
do if (legmom=1 or legdad=1) and (ythcitiz=-4).  
    if (p2001=1) ythcitiz=1.  
    if (p2001 ne 1) and (p2008<=dob01y) ythcitiz=3.
```

```
        if (p2001 ne 1) and (p2008>dob01y) ythcitiz=2.  
end if.  
do if (splegmom=1 or splegdad=1) and (ythcitiz ne 1).  
    if (p2037=1) ythcitiz=1.  
    if (p2037 ne 1) and (p2044<=dob01y) ythcitiz=3.  
    if (p2037 ne 1) and (p2044>dob01y) ythcitiz=2.  
end if.  
do if (legmom ne 1 and legdad ne 1) and (ythcitiz ne 1).  
    if (p210101=1 or p210102=1) ythcitiz=1.  
    if (p210101 ne 1) and (p2108b01<=dob01y and  
        p2108b01>0) ythcitiz=3.  
    if (p210101 ne 1) and (p2108b01>dob01y)  
        ythcitiz=2.  
    if (p210101<0) and (p210102<0) and  
        (p2108b01<0) ythcitiz=3.  
end if.  
if (pinf015y=-4) ythcitiz=-4.  
  
*****hand edits using youth roster data when  
paryouth data missing*****  
if (pubid=1745 or pubid=5812 or pubid=6481 or  
    pubid=6482 or pubid=6483 or pubid=6748 or  
    pubid=6749) ythcitiz=3.  
if (pubid=4659 or pubid=4679 or pubid=7358)  
    ythcitiz=3.  
  
execute.
```

YOUTH'S RELATIONSHIP TO HOUSEHOLD PARENT FIGURE(S) AT AGE 2, 6, AND 12

Variables Created:

CV_HH_REL_AGE_2
CV_HH_REL_AGE_6
CV_HH_REL_AGE_12

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
PUBID	PUBID	P818200-13	PC8-018.02_001-_014
dob01d, m, y	KEY!BDATE_D,_M,_Y	P818300-13	PC8-018.03_001-_014
P3051Y1-P3051Y6	P3-051.01_Y-.06_Y	P818400-13	PC8-018.04_001-_014
P305301-P305306	P3-053.01-.06	P818500-13	PC8-018.05_001-_014
P3067Y1-P3067Y6	P3-067.01_Y-.06_Y	P818600-13	PC8-018.06_001-_014
P3067Y2-5	P3-067.02_Y-.05_Y	P818700-13	PC8-018.07_001-_014
P818100-13	PC8-018.01_001-_014	P818800-13	PC8-018.08_001-_014
P8181000-13	PC8-018.10_001-_014	P818900-13	PC8-018.09_001-_014
P8181100-13	PC8-018.11_001-_014	PC801901-17	PC8-019.01-17
P8181200-13	PC8-018.12_001-_014	PC802001-16	PC8-020.01-16
P8181300-13	PC8-018.13_001-_014	PC802101-16	PC8-021.01-16
P8181400-13	PC8-018.14_001-_014	PC802201-16	PC8-022.01-16
P8181500-13	PC8-018.15_001-_014	PC803101-11	PC8-031.01-11
P8181600-13	PC8-018.16_001-_014	PC803201-11	PC8-032.01-11
P8181700-13	PC8-018.17_001-_014	PC803501-10	PC8-035.01-10

Codes for Created Variable

1 = Both biological parents

2 = Biological mother, other parent present

3 = Biological father, other parent present

4 = Biological mother, marital status unknown

5 = Biological dad, marital status unknown

6 = Adoptive parent(s)

7 = Foster parent(s)

8 = Other adults, biological parent status unknown, not group quarters

9 = Group quarters

10 = Anything else

This program creates variables indicating the relationship of the youth to the primary adults in the household when the youth was age 2, 6, and 12.

```

array yrelate(3) yrelate2 yrelate6 yrelat12;
array continue(6) p305301-p305306;
array maryear(6) p3051y1-p3051y6;
array stopyear(6) p3067y1-p3067y6;
array stoplive(16) pc802201-pc802216;
array livewh(17, 14) p818100-p818113 p818200-p818213 p818300-p818313 p818400-p818413 p818500-
    p818513 p818600-p818613 p818700-p818713 p818800-p818813 p818900-p818913 p8181000-
    p8181013 p8181100-p8181113 p8181200-p8181213 p8181300-p8181313 p8181400-p8181413
    p8181500-p8181513 p8181600-p8181613 p8181700-p8181713;
array awayyear(10) pc803501-pc803510;
array awaywho(11) pc803201-pc803211;
array loopnum(3) loopnum2 loopnum6 loopnm12;
array respondt(17) pc801901-pc801917;
array noknow(17) pc802001-pc802017;
array refuse(17) pc802101-pc802117;
array place(11) pc803101-pc803111;
array lastnum(3) lastnum1 lastnum2 lastnum3;

/***three loops because of 2, 6, 12***/ 
do i= 1 to 3;

```

Appendix 3: Family Background Variable Creation

```
if i=1 then age=dob01y + 2;
else if i=2 then age=dob01y +6;
else if i=3 then age=dob01y +12;

/** Initialize everyone to zero or -4 **/
yrelate(i)=0;
loopnum(i)=0;

/* 1. The youth lived prior to live with parents */

if age > 1900 then do;

/**cases satisfied the following condition should have yrelate > 0. For age 2, the exception is those exiting from the first loop and stay with respondent or don't know or refuse this information; don't know or refuse in year first lived together; and don't know or refused the year relationship ended. If the year of the last loop larger than 0 but less than age then exit condition with the respondent then compute in the following sections. These cases still have yrelate=0 after this section and will be processed in section 2 or section 3. The same process is followed for age 6 and 12.**/

if pc8009y1 >=age or pc8009y1 in (-1,-2) then do;

****a. identify the loop number into which age 2 (6,12) falls****
/**note that if there are multiple changes in the age year then the loopnum takes the last one**/
/**there are 15 loops because of the 'j+1,' and the 17th loop is the respondent**/
do j=1 to 15;
  if stoplive(1) >=age then loopnum(i)=1;
  if 0 < stoplive(j) < age and stoplive(j+1) >= age then loopnum(i)=j+1;
/**the next line ensures that the last relationship reported during the age is the one used**/
  if stoplive(j)=stoplive(j+1)=age then loopnum(i)=j+1;
end;

/**above deals with the 'good' data; below deals with the 'obnoxious' data**/
/**max year of the prior loops**/
maxyear=max(of pc802201.pc802216);

/** taking care of the cases for pc8009y1 (year first lived)>=age and loopnum(i)=0 (exit from the first loop, age does not fall into any interval due to -1, -2 in stoplive or errors in reported date)**/
if loopnum(i)=0 and pc8009y1 >=age then do;
  /**note that if the loop ended in the first loop, then maxyear=-4, loopnum(i)=0;**/
  /**the situation that maxyear=-4 and respondt(1)=1 (exit condition for 1st loop) will be taken care of later; takes care of the dk, refuse first**/
  if maxyear=-4 and noknow(1)=1 then loopnum(i)=-2;
  if maxyear=-4 and refuse(1)=1 then loopnum(i)=-1;
/**reported error - dont know or refuse the year or give 'wrong' year (inconsistency)**/
  if maxyear in (-1,-2) then loopnum(i)=maxyear;
  else if 0 < maxyear < age then loopnum(i)=-1 ;
  else if maxyear >=age then do k=1 to 16;
    if stoplive(k) in (-1,-2) and loopnum(i) ^=-1 then loopnum(i)=stoplive(k);
  end;
end;

/** taking care of the cases when pc8009y1 is -1 (-2) and loopnum=0**/
if loopnum(i)=0 and pc8009y1 in (-1,-2) then do;
  if maxyear in (-1,-2,-4) and (pc802601 >=age or pc802601 in (-1,-2,-4))
    then loopnum(i)=pc8009y1;  /**for those with pc802601< age, loopnum=0(yrelate=0)**/
  if maxyear >=age then loopnum(i)=pc8009y1;
```

```

        counter1=0;
/*this determines the last loop reported by a respondent, regardless of whether the data are real, dont know or
refuse***/ 

if 0 < maxyear < age then do;
do j=1 to 16;    /***find out which loop is the last one ***/
if stoplive(j)=-4 and counter1=0 then do;
  lastloop=j-1;      counter1=counter1+1;
end;
end;
if j=16 and stoplive(16) ^=-4 then lastloop=16;

*****if stoplive(lastloop)>0 (valid year) and respondt(lastloop+1)=1  computed in one of the following sections -
2 or 3;
  if stoplive(lastloop)>0 and (noknow(lastloop+1)=1 or refuse(lastloop+1)=1) then loopnum(i)=pc8009y1;
  if stoplive(lastloop) in (-1,-2) then loopnum(i)=pc8009y1;
  end;
end;

/*b. recode the yrelate***/ 

if loopnum(i) in (-1,-2) then yrelate(i)=loopnum(i);

if loopnum(i) > 0 then do;
/** lived with bio. mom, marital status unknown **/
if livewh(loopnum(i),2 )=1 then yrelate(i)=4;

/* lived with bio dad, marital status unknown*/
if livewh(loopnum(i),3 )=1 then yrelate(i)=5;

/* lived with both biological parents*/
if livewh(loopnum(i),2 )=1 and livewh(loopnum(i),3)=1 then yrelate(i)=1;

/* lived with adoptive parents*/
if livewh(loopnum(i),4 )=1 and livewh(loopnum(i),5)=1 then yrelate(i)=8;

/* lived with other adults, bio. parent status unknown, not group qtrs */
if livewh(loopnum(i),4 )=1 then yrelate(i)=8;
if livewh(loopnum(i),5 )=1 then yrelate(i)=8;

/* lived with foster parents*/
if livewh(loopnum(i),7 )=1 then yrelate(i)=7;

/*lived in group quarters*/
do k = 8,9,10,11,12;
  if livewh(loopnum(i),k )=1 then yrelate(i)=9;
end;

/* any other living arrangement*/
do k = 6,13,14;
  if livewh(loopnum(i),k )=1 then yrelate(i)=10;
end;
end;
end;

/* 2. check for periods spent away from parents, pc802601 <age. Note that if the youth lived with respondent at
age 2, 6, 12, then yrelate=0. These cases will be coded in section 3. */

```

Appendix 3: Family Background Variable Creation

```
if (pc802501=1) and (pc802601 < age ) and yrelate(i)=0 then do;  
/****note that if pc802601=-1 or -2, then the case is included****/  
  
/***find out which loop is the last one; there is no change after that; think of counter 2 as a flag ***/  
counter2=0;  
do j=1 to 10;  
if awayyear(j)=-4 and counter2=0 then do;  
    lastchg=j-1;  
    if j=1 then do;  
        lastchg=1;  
        nochange=1;  
        end;  
    counter2=counter2+1;  
end;  
end;  
if awayyear(10)>-4 then lastchg=10;  
  
/***find out into which loop the age falls ***/  
do j=1 to lastchg;      /***10 equals to the maximum complete spells away from parents***/  
if j=1 and awayyear(1)=-4 then lastnum(i)=1;  
if j=1 and awayyear(1) >= age then lastnum(i)=1;  
if j < lastchg and 0< awayyear(j) < age and awayyear(j+1) >= age then lastnum(i)=j+1;  
if j < lastchg and awayyear(j)=awayyear(j+1)=age then lastnum(i)=j+1;  
if j=lastchg and 0 < awayyear(lastchg)< age then lastnum(i)=lastchg+1;  
end;  
  
if lastnum(i)=-1 then lastnum(i)=-1; /*if there is -1/-2 in awayyear, no such cases in current data***/  
  
/***r goes on to give more data, but the year info in the first loop is invalid***/  
if lastnum(i)=1 and nochange ^= 1 and pc802601 in (-1,-2) then yrelate(i)=pc802601;  
/***yrelate=-1 or -2***/  
if lastnum(i)=-1 then yrelate(i)=-1;  
if yrelate(i) =0 then do;  
    if place(lastnum(i)) in (-1,-2) then yrelate(i)=place(lastnum(i));  
    if place(lastnum(i)) in (1,4) then yrelate(i)=10; /***coded as 'anything else'***/  
    if place(lastnum(i))=3 then yrelate(i)=9;      /***group quarters***/  
    if place(lastnum(i))=2 then do;  
/**if reponent (1) is coded [awaywho(lastnum(i))=1] then go to section 3 to determine marital status**/  
        if awaywho(lastnum(i)) = 2 then yrelate(i)=1;  
        if awaywho(lastnum(i)) = 3 then yrelate(i)=4;  
        if awaywho(lastnum(i)) = 4 then yrelate(i)=5;  
        if awaywho(lastnum(i)) in (5,6,11) then yrelate(i)=8;  
        if awaywho(lastnum(i)) in (7,8,9) then yrelate(i)=10;  
        if awaywho(lastnum(i)) =10 then yrelate(i)=7;  
        if awaywho(lastnum(i))in (-1,-2) then yrelate(i)=awaywho(lastnum(i));  
    end;  
end;  
end;  
  
/* 3. The youth lived with parents continuously or the first separation is >= age or the separation time is with  
respondent. Flag2 = 1 indicates that a marriage happened before or when youth is 2 (6,12) and ended when or after  
youth is 2 */  
  
if yrelate(i)=0 and pc802501 in (-1,-2) then yrelate(i)=pc802501;  
if yrelate(i)=0 and ( (pc802501=0) or (pc802501=1 and pc802601 >= age) or
```

Appendix 3: Family Background Variable Creation

```
( (pc802501=1) and (pc802601 < age) ) then do;
do j = 1 to 6;*** Determine if the youth has lived with both parents since living with the respondent ***
  if continue(j)=1 and ( 0 <= dob01y - maryear(j) < 1900 ) and ythpar01 in (1,2) and yth2bios=1 then do;
    yrelate(i)=1;
  end;
end;

if yrelate(i)=0 then do;
  if max(of p3051y1-p3051y6)=-4 then flag1=-4;
  do j=1 to 6;
    if j = < 5 and (1900 < maryear(j) < age) and (stopyear(j) >=age) then flag1=1;
    else if j=6 and (1900 < maryear(j) < age) then flag1=1;
  end;
  if flag1=. then flag1=-1; /*for those with -1,-2 in the age period or the age is not with any marriages */
  if flag1=1 then do;
    if ythpar01=1      then yrelate(i)=2;           *** bio. mom, other parent figure present ***
    else if ythpar01=2  then yrelate(i)=3;           *** bio. dad, other parent figure present ***
    end;
  else if flag1 in (-1,-4) then do;
    if ythpar01=1 then yrelate(i)=4;
    else if ythpar01=2 then yrelate(i)=5;
  end;
  if ythpar01 in (3,4) then yrelate(i)=6;           *** adoptive parents ***
  else if ythpar01 in (5,7,11,15,17,19,21,6,8,12,16,18,20,22,99) then yrelate(i)=8;
    *** other adults, bio. parent status unknown ***
  else if ythpar01 in (9,13,10,14) then yrelate(i)=7; *** foster/temporary parents ***
  end;
end;

end;
end;

if pc800401=-4 then yrelate2=-4;
if pc800401=-4 then yrelate6=-4;
if pc800401=-4 then yrelate12=-4;

*hand edits;
if pubid=228 then yrelate2=5 and yrelate6=5 and yrelate12=5;
if pubid=1075 then yrelate2=8 and yrelate6=8 and yrelate12=8;
if pubid=1128 then yrelate2=4 and yrelate6=4 and yrelate12=4;
if pubid=3404 then yrelate2=1 and yrelate6=1 and yrelate12=1;
if pubid=8020 then yrelate2=5 and yrelate6=5 and yrelate12=5;

endsas;
```

NLSY97 Appendix 4:
Geographic Variable Creation

Several variables in the main data set provide information about the respondent's area of residence. These variables are intended to permit researchers to identify key characteristics of the area without needing access to the geocode CD-ROM. Geographic variables were created using a software program called Matchmaker for Windows, V2.5; therefore, no programming code is provided for these variables. Instead, this document offers a brief general description of the methods used to generate these variables. For more information about the process of classifying a respondent's metropolitan area or about the geographic variables in general, refer to the introduction to the *Geocode Codebook Supplement* or contact NLS User Services.

CENSUS REGION OF RESIDENCE AT SURVEY DATE AND AT AGE 12

Variables Created: CV_CENSUS_REGION
CV_CENSUS_REGION_AGE_12

These variables classify respondents as residing in one of four regions defined by the U.S. Bureau of the Census. These regions are as follows:

Census Division	States
Northeast	Connecticut, Maine, Massachusetts, New Hampshire, New Jersey, New York, Pennsylvania, Rhode Island, and Vermont
North Central	Illinois, Indiana, Iowa, Kansas, Michigan, Minnesota, Missouri, Nebraska, North Dakota, Ohio, South Dakota, and Wisconsin
South	Alabama, Arkansas, Delaware, District of Columbia, Florida, Georgia, Kentucky, Louisiana, Maryland, Mississippi, North Carolina, Oklahoma, South Carolina, Tennessee, Texas, Virginia, and West Virginia
West	Alaska, Arizona, California, Colorado, Hawaii, Idaho, Montana, Nevada, New Mexico, Oregon, Utah, Washington, Wyoming

MSA STATUS AT SURVEY DATE AND AT AGE 12

Variables Created: CV_MSA
CV_MSA_AGE_12

This variable provides users with information about whether the respondent lived in the central city of the MSA, in another part of the MSA, or outside of an MSA. As defined by the Census Bureau, a central city is the major city lying within a Metropolitan Statistical Area (MSA). Initially, two variables were created using the TIGER/Line files (a database developed by the Census Bureau) to classify respondent residences: one identifying the respondent's MSA status and one identifying the respondent's central city status. The variables were then combined to produce a single MSA/central city variable. For this dataset, respondents are coded as follows:

- 1 not in MSA
- 2 in MSA, not central city
- 3 in MSA, central city
- 4 in MSA, not known
- 5 not in country

RURAL VS. URBAN

Variables Created: CV_URBAN_RURAL
CV_URBAN_RURAL AGE_12

As defined by the geocode software, urban places are “closely settled, named, communities that generally contain a mixture of residential, commercial, and retail areas, and have a population greater than 2,500.” These criteria are based on those used by the Census Bureau to identify urban areas. All other places are considered rural. Based on the TIGER/Line files, respondents residing in urban areas are coded 1 and those residing in rural areas are coded 0. Census Bureau information on urban and rural places can be retrieved from the following internet site:

<http://www.census.gov/geo/www/tiger/index.html>

NLSY97 Appendix 5:
Income and Assets Variable Creation

HOUSEHOLD INCOME AND ASSETS

Variables Created:	CV_HH_NET_WORTH_P CV_INCOME_GROSS_YR CV_HH_POV_RATIO	CV_HH_NET_WORTH_Y CV_HH_INCOME_SOURCE
---------------------------	--	--

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
YI_300	YINC-300	YI_17900	YINC-17900
YI_400	YINC-400	YI_18000	YINC-18000
YI_1400 – YI2700	YINC-1400 – YINC-2700	YI_18100 – YI_19900	YINC-18100 – YINC-19900
YI_2900	YINC-2900	YI_20000 – YI_20500	YINC-20000 – YINC-20500
YI_3000	YINC-3000	YI_20800	YINC-20800
YI_3100	YINC-3100	I2100001–20	YINC-21000_001–_020
YI_3300	YINC-3300	I2140001–20	YINC-21400.01–.20
YI_3400	YINC-3400	I2170001–20	YINC-21700.01–.20
YI_3900 – YI_5400	YINC-3900 – YINC-5400	I2180001–20	YINC-21800.01–.20
YI_55001–07	YINC-5500_001–5500_007	I2190009	YINC-21900.09
YI_5600 – YI_6300	YINC-5600 – YINC-6300	YI_22100 – YI_22300	YINC-22100 – YINC-22300
YI_6310	YINC-6310	PUBID	PUBID
YI_6320	YINC-6320	P5_010	P5-010
YI_6400 – YI_6900	YINC-6400 – YINC-6900	P5_011	P5-011
YI_6910 – YI_6940	YINC-6910 – YINC-6940	P5_012	P5-012
YI_7000 – YI_7500	YINC-7000 – YINC-7500	P5_016 – P5_023	P5-016 – P5-023
YI_7510 – YI_7540	YINC-7510 – YINC-7540	P5_028	P5-028
YI_7600 – YI_9900	YINC-7600 – YINC-9900	P5_029	P5-029
YI_10000 – YI_10700	YINC-10000 – YINC-10700	P5_031 – P5_033	P5031 – P5-033
I111001–09	YINC-11100.01–09	P5_045 – P5_050	P5-045 – P5-050
I111101–08	YINC-11110.01–08	P5_052 – P5_057	P5-052 – P5-057
I112001–08	YINC-11200.01–08	P5_063 – P5_069	P5-063 – P5-069
I116001–08	YINC-11600.01–08	P5_071	P5-071
I117001–08	YINC-11700.01–08	P5_07301–09	P5-073.01–.09
YI_12200 – YI_15200	YINC-12200 – YINC-15200	P5_07401–09	P5-074.01–.09
YI_15800 – YI_16700	YINC-15800 – YINC-16700	P5_07701–09	P5-077.01–.09
I1680001–20	YINC-16800.01–20	P5_07801–09	P5-078.01–.09
YI_16900	YINC-16900	P5_082 – P5_099	P5-082 – P5-099
YI_17000	YINC-17000	P5_100 – P5_105	P5-100 – P5_105
YI_17100	YINC-17100	P5_112 – P5_122	P5-112 – P5-122
YI_17200	YINC-17200	P5_124 – P5_156	P5-124 – P5-156
YI_17300	YINC-17300	PNORCID	PNORCID
YI_17400	YINC-17400	KAGE	KEY!AGE
YI_17500	YINC-17500	PUBID	PUBID
YI_17600	YINC-17600	PINF20	PINF-020
YI_17700	YINC-17700	PARYTHID	PARYOUTH_ID
YI_17800	YINC-17800	PARYTHR	PARYOUTH_PARENT

This program creates the Household Net Worth and Gross Household Income variables (according to information collected from the parent and youth). The household net worth variables according to the parent (hhworthp) and to the youth (hhworthY) are actual numbers that result from taking all assets and subtracting liabilities of the household. The gross household income according to the parent (groshhIp) and to the youth (groshhIY) includes total annual cash receipts before taxes from all sources. An indicator variable is also created to record whether the household income was according to the parent or youth. Finally, a variable is created indicating the ratio of household income to the poverty level.

******SECTION 1: PARENT HOUSEHOLD NET WORTH AND GROSS HOUSEHOLD INCOME ****/**

flag=0;

Appendix 5: Income and Assets Variable Creation

```
/*First, create a variable indicating net receipts from non-farm employment earned by the R, such as wages  
(nfarmwgR). */  
nfarmwgR=0;  
if (P5_010=1 or P5_011=1) and P5_016 not in (-1, -2, -3) then nfarmwgR=P5_016;  
if (P5_010=1 or P5_011=1) and (P5_016 eq -1 or P5_016 eq -2 or P5_016 eq -3) then do;  
    if P5_017=1 then do; nfarmwgR=2500; flag=1; end;  
    if P5_017=2 then do; nfarmwgR=7500; flag=1; end;  
    if P5_017=3 then do; nfarmwgR=17500; flag=1; end;  
    if P5_017=4 then do; nfarmwgR=37500; flag=1; end;  
    if P5_017=5 then do; nfarmwgR=75000; flag=1; end;  
    if P5_017=6 then do; nfarmwgR=175000; flag=1; end;  
    if P5_017=7 then do; nfarmwgR=250001; flag=1; end;  
    end;  
if P5_011=-2 or P5_017=-2 then nfarmwgR=-2;  
if P5_012=-1 or P5_017=-1 then nfarmwgR=-1;  
if P5_011=-3 or P5_012=-3 or P5_017=-3 then nfarmwgR=-3;  
  
/*Second, create a variable indicating net receipts from farm self-employment earned by the R (farmwgR)*/  
farmwgR=0;  
if P5_018=1 and P5_019 not in (-1, -2, -3) then farmwgR=P5_019;  
if P5_018=1 and (P5_019 eq -1 or P5_019 eq -2 or P5_019 eq -3) then do;  
    if P5_020=1 then do; farmwgR=-2; flag=1; end;  
    if P5_020=2 then do; farmwgR=2500; flag=1; end;  
    if P5_020=3 then do; farmwgR=7500; flag=1; end;  
    if P5_020=4 then do; farmwgR=17500; flag=1; end;  
    if P5_020=5 then do; farmwgR=37500; flag=1; end;  
    if P5_020=6 then do; farmwgR=75000; flag=1; end;  
    if P5_020=7 then do; farmwgR=175000; flag=1; end;  
    if P5_020=8 then do; farmwgR=250001; flag=1; end;  
    end;  
if P5_018=-1 or P5_020=-1 then farmwgR=-1;  
if P5_018=-2 or P5_020=-2 then farmwgR=-2;  
if P5_018=-3 or P5_020=-3 then farmwgR=-3;  
  
/*Third, create the above variables for the R's spouse/partner: nfarmwgS and farmwgS respectively */  
nfarmwgS=0;  
if P5_021=1 and P5_022=1 and P5_028 not in (-1, -2, -3) then nfarmwgS=P5_028;  
if P5_021=1 and P5_022=1 and (P5_028 eq -1 or P5_028 eq -2 or P5_028 eq -3) then do;  
    if P5_029=1 then do; nfarmwgS=2500; flag=1; end;  
    if P5_029=2 then do; nfarmwgS=7500; flag=1; end;  
    if P5_029=3 then do; nfarmwgS=17500; flag=1; end;  
    if P5_029=4 then do; nfarmwgS=37500; flag=1; end;  
    if P5_029=5 then do; nfarmwgS=75000; flag=1; end;  
    if P5_029=6 then do; nfarmwgS=175000; flag=1; end;  
    if P5_029=7 then do; nfarmwgS=250001; flag=1; end;  
    end;  
if P5_022=-1 or P5_029=-1 then nfarmwgS=-1;  
if P5_022=-2 or P5_029=-2 then nfarmwgS=-2;  
if P5_022=-3 or P5_029=-3 then nfarmwgS=-3;  
  
farmwgS=0;  
if P5_021=1 and P5_031=1 and P5_032 not in (-1, -2, -3) then farmwgS=P5_032;  
if P5_021=1 and P5_031=1 and (P5_032 eq -1 or P5_032 eq -2 or P5_032 eq -3) then do;  
    if P5_033=1 then do; farmwgS=-2; flag=1; end;  
    if P5_033=2 then do; farmwgS=2500; flag=1; end;  
    if P5_033=3 then do; farmwgS=7500; flag=1; end;  
    if P5_033=4 then do; farmwgS=17500; flag=1; end;
```

```

if P5_033=5 then do; farmwgS=37500; flag=1; end;
if P5_033=6 then do; farmwgS=75000; flag=1; end;
if P5_033=7 then do; farmwgS=175000; flag=1; end;
if P5_033=8 then do; farmwgS=250001; flag=1; end;
end;
if P5_031=-1 or P5_033=-1 then farmwgS=-1;
if P5_031=-2 or P5_033=-2 then farmwgS=-2;
if P5_031=-3 or P5_033=-3 then farmwgS=-3;

/*Fourth, create a variable capturing the earned interests by the R and spouse (S) from various accounts */
interest=0;
if P5_045=1 and P5_046 not in (-1, -2, -3) then interest=P5_046;
if P5_045=1 and (P5_046 eq -1 or P5_046 eq -2 or P5_046 eq -3) then do;
    if P5_047=1 then do; interest=500; flag=1; end;
    if P5_047=2 then do; interest=1750; flag=1; end;
    if P5_047=3 then do; interest=3750; flag=1; end;
    if P5_047=4 then do; interest=7500; flag=1; end;
    if P5_047=5 then do; interest=17500; flag=1; end;
    if P5_047=6 then do; interest=37500; flag=1; end;
    if P5_047=7 then do; interest=50001; flag=1; end;
end;
if P5_045 eq -1 or P5_047=-1 then interest=-1;
if P5_045 eq -2 or P5_047=-2 then interest=-2;
if P5_045 eq -3 or P5_047=-3 then interest=-3;

/*Fifth, create a variable capturing any received AFDC or ADC */
AFDCADC=0;
if P5_048=1 and P5_049 not in (-1, -2, -3) then AFDCADC=P5_049;
if P5_048=1 and (P5_049 eq -1 or P5_049 eq -2 or P5_049 eq -3) then do;
    if P5_050=1 then do; AFDCADC=250; flag=1; end;
    if P5_050=2 then do; AFDCADC=750; flag=1; end;
    if P5_050=3 then do; AFDCADC=1750; flag=1; end;
    if P5_050=4 then do; AFDCADC=3750; flag=1; end;
    if P5_050=5 then do; AFDCADC=6250; flag=1; end;
    if P5_050=6 then do; AFDCADC=8750; flag=1; end;
    if P5_050=7 then do; AFDCADC=10001; flag=1; end;
end;
if P5_048=-1 or P5_050=-1 then AFDCADC=-1;
if P5_048=-2 or P5_050=-2 then AFDCADC=-2;
if P5_048=-3 or P5_050=-3 then AFDCADC=-3;

/*Sixth, create a variable indicating whether any food stamps are collected */
foodst=0;
if P5_052=1 and P5_053 not in (-1, -2, -3) then foodst=P5_053;
if P5_052=1 and (P5_053 eq -1 or P5_053 eq -2 or P5_053 eq -3) then do;
    if P5_054=1 then do; foodst=250; flag=1; end;
    if P5_054=2 then do; foodst=750; flag=1; end;
    if P5_054=3 then do; foodst=1750; flag=1; end;
    if P5_054=4 then do; foodst=3750; flag=1; end;
    if P5_054=5 then do; foodst=6250; flag=1; end;
    if P5_054=6 then do; foodst=8750; flag=1; end;
    if P5_054=7 then do; foodst=10001; flag=1; end;
end;
if P5_052=-1 or P5_054=-1 then foodst=-1;
if P5_052=-2 or P5_054=-2 then foodst=-2;
if P5_052=-3 or P5_054=-3 then foodst=-3;

```

Appendix 5: Income and Assets Variable Creation

```
/*Seventh, create a variable indicating whether the R and S received any SSI */
SSI=0;
if P5_055=1 and P5_056 not in (-1, -2, -3) then SSI=P5_056;
if P5_055=1 and (P5_056 eq -1 or P5_056 eq -2 or P5_056 eq -3) then do;
    if P5_057=1 then do; SI=250; flag=1; end;
    if P5_057=2 then do; SSI=750; flag=1; end;
    if P5_057=3 then do; SSI=1750; flag=1; end;
    if P5_057=4 then do; SSI=3750; flag=1; end;
    if P5_057=5 then do; SSI=6250; flag=1; end;
    if P5_057=6 then do; SSI=8750; flag=1; end;
    if P5_057=7 then do; SSI=10001; flag=1; end;
    end;
if P5_055=-1 or P5_057=-1 then SSI=-1;
if P5_055=-2 or P5_057=-2 then SSI=-2;
if P5_055=-3 or P5_057=-3 then SSI=-3;

/*Eighth, create a variable indicating whether they collected any child support */
childsup=0;
if P5_064=1 and P5_065 not in (-1, -2, -3) then childsup=P5_065;
if P5_064=1 and (P5_065 eq -1 or P5_065 eq -2 or P5_065 eq -3) then do;
    if P5_066=1 then do; childsup=500; flag=1; end;
    if P5_066=2 then do; childsup=1750; flag=1; end;
    if P5_066=3 then do; childsup=3750; flag=1; end;
    if P5_066=4 then do; childsup=7500; flag=1; end;
    if P5_066=5 then do; childsup=17500; flag=1; end;
    if P5_066=6 then do; childsup=37500; flag=1; end;
    if P5_066=7 then do; childsup=50001; flag=1; end;
    end;
if P5_064=-1 or P5_066=-1 then childsup=-1;
if P5_064=-2 or P5_066=-2 then childsup=-2;
if P5_064=-3 or P5_066=-3 then childsup=-3;

/* Next create another variable capturing any rental income, as well as workers' compensation benefits,
welfare and unemployment benefits, from the R and S */
rentalI=0;
if P5_067=1 and P5_068 not in (-1, -2, -3) then rentalI=P5_068;
if P5_067=1 and (P5_068 eq -1 or P5_068 eq -2 or P5_068 eq -3) then do;
    if P5_069=1 then do; rentalI=500; flag=1; end;
    if P5_069=2 then do; rentalI=1750; flag=1; end;
    if P5_069=3 then do; rentalI=3750; flag=1; end;
    if P5_069=4 then do; rentalI=7500; flag=1; end;
    if P5_069=5 then do; rentalI=17500; flag=1; end;
    if P5_069=6 then do; rentalI=37500; flag=1; end;
    if P5_069=7 then do; rentalI=50001; flag=1; end;
    end;
if P5_067=-1 or P5_069=-1 then rentalI=-1;
if P5_067=-2 or P5_069=-2 then rentalI=-2;
if P5_067=-3 or P5_069=-3 then rentalI=-3;

/*Now a variable capturing any income from any family member older than 14 other than R and spouse (S) */
array famI famI01 famI02 famI03 famI04 famI05 famI06 famI07 famI08 famI09;
array P5_073 P5_07301 P5_07302 P5_07303 P5_07304 P5_07305 P5_07306 P5_07307 P5_07308 P5_07309;
array P5_074 P5_07401 P5_07402 P5_07403 P5_07404 P5_07405 P5_07406 P5_07407 P5_07408 P5_07409;
array P5_077 P5_07701 P5_07702 P5_07703 P5_07704 P5_07705 P5_07706 P5_07707 P5_07708 P5_07709;
array P5_078 P5_07801 P5_07802 P5_07803 P5_07804 P5_07805 P5_07806 P5_07807 P5_07808 P5_07809;

do I=1 to 9;
```

```

famI(I)=0;
if P5_073(I)=0 and P5_074(I)=0 then famI(I)=(P5_077(I)+P5_078(I));
if P5_073(I)=-1 or P5_074(I)=-1 or P5_077(I)=-1 or P5_078(I)=-1 then famI(I)=-1;
if P5_073(I)=-2 or P5_074(I)=-2 or P5_077(I)=-2 or P5_078(I)=-2 then famI(I)=-2;
if P5_073(I)=-3 or P5_074(I)=-3 or P5_077(I)=-3 or P5_078(I)=-3 then famI(I)=-3;
end;

/*A variable indicating the value of the property if the R lives in a ranch or farm */
pvranch=0;
if P5_082=1 and P5_084=1 and P5_086 not in (-1, -2, -3) then pvranch=P5_086;
if P5_082=1 and P5_084=1 and (P5_086 eq -1 or P5_086 eq -2 or P5_086 eq -3) then do;
    if P5_087=1 then do; pvranch=12500; flag=1; end;
    if P5_087=2 then do; pvranch=37500; flag=1; end;
    if P5_087=3 then do; pvranch=75000; flag=1; end;
    if P5_087=4 then do; pvranch=175000; flag=1; end;
    if P5_087=5 then do; pvranch=375000; flag=1; end;
    if P5_087=6 then do; pvranch=750000; flag=1; end;
    if P5_087=7 then do; pvranch=1000001; flag=1; end;
    end;
if P5_082=1 and P5_084=2 and P5_088 not in (-1, -2, -3) and P5_091 not in (-1, -2, -3) then pvranch=P5_091;
if P5_082=1 and P5_084=2 and P5_088 not in (-1, -2, -3) and (P5_091= -1 | P5_091= -2 | P5_091= -3) then do;
    if P5_092=1 then do; pvranch=12500; flag=1; end;
    if P5_092=2 then do; pvranch=37500; flag=1; end;
    if P5_092=3 then do; pvranch=75000; flag=1; end;
    if P5_092=4 then do; pvranch=175000; flag=1; end;
    if P5_092=5 then do; pvranch=375000; flag=1; end;
    if P5_092=6 then do; pvranch=750000; flag=1; end;
    if P5_092=7 then do; pvranch=1000001; flag=1; end;
    end;
if P5_082=1 and P5_084=2 and P5_088 ge 1 and P5_089 not in (-1, -2, -3) then pvranch=P5_089;
if P5_082=1 and P5_084=2 and P5_088 ge 1 and (P5_089 eq -1 or P5_089 eq -2 or P5_089 eq -3) then do;
    if P5_090=1 then do; pvranch=12500; flag=1; end;
    if P5_090=2 then do; pvranch=37500; flag=1; end;
    if P5_090=3 then do; pvranch=75000; flag=1; end;
    if P5_090=4 then do; pvranch=175000; flag=1; end;
    if P5_090=5 then do; pvranch=375000; flag=1; end;
    if P5_090=6 then do; pvranch=750000; flag=1; end;
    if P5_090=7 then do; pvranch=1000001; flag=1; end;
    end;
if P5_082=-1 or P5_084=-1 or P5_087=-1 or P5_090=-1 or P5_092=-1 then pvranch=-1;
if P5_082=-2 or P5_084=-2 or P5_087=-2 or P5_090=-2 or P5_092=-2 then pvranch=-2;
if P5_082=-3 or P5_084=-3 or P5_087=-3 or P5_090=-3 or P5_092=-3 then pvranch=-3;

/*A variable indicating the value of the property if the R lives in a mobile home */
pvmobile=0;
if P5_083=1 and P5_093=1 and P5_095 not in (-1, -2, -3) then pvmobile=P5_095;
if P5_083=1 and P5_093=1 and (P5_095 eq -1 or P5_095 eq -2 or P5_095 eq -3) then do;
    if P5_096=1 then do; pvmobile=12500; flag=1; end;
    if P5_096=2 then do; pvmobile=37500; flag=1; end;
    if P5_096=3 then do; pvmobile=75000; flag=1; end;
    if P5_096=4 then do; pvmobile=175000; flag=1; end;
    if P5_096=5 then do; pvmobile=375000; flag=1; end;
    if P5_096=6 then do; pvmobile=750000; flag=1; end;
    if P5_096=7 then do; pvmobile=1000001; flag=1; end;
    end;
if P5_083=1 and P5_093=2 and P5_097 not in (-1, -2, -3) then pvmobile=P5_097;
if P5_083=1 and P5_093=2 and (P5_097 eq -1 or P5_097 eq -2 or P5_097 eq -3) then do;

```

```

if P5_098=1 then do; pvmobile=2500; flag=1; end;
if P5_098=2 then do; pvmobile=7500; flag=1; end;
if P5_098=3 then do; pvmobile=17500; flag=1; end;
if P5_098=4 then do; pvmobile=37500; flag=1; end;
if P5_098=5 then do; pvmobile=75000; flag=1; end;
if P5_098=6 then do; pvmobile=175000; flag=1; end;
if P5_098=7 then do; pvmobile=250001; flag=1; end;
end;

if P5_083=1 and P5_093=3 and P5_099 not in (-1, -2, -3) then pvmobile=P5_099;
if P5_083=1 and P5_093=3 and (P5_099 eq -1 or P5_099 eq -2 or P5_099 eq -3) then do;
    if P5_100=1 then do; pvmobile=2500; flag=1; end;
    if P5_100=2 then do; pvmobile=7500; flag=1; end;
    if P5_100=3 then do; pvmobile=17500; flag=1; end;
    if P5_100=4 then do; pvmobile=37500; flag=1; end;
    if P5_100=5 then do; pvmobile=75000; flag=1; end;
    if P5_100=6 then do; pvmobile=175000; flag=1; end;
    if P5_100=7 then do; pvmobile=250001; flag=1; end;
end;

if P5_083=-1 or P5_093=-1 or P5_096=-1 or P5_098=-1 or P5_100=-1 then pvmobile=-1;
if P5_083=-2 or P5_093=-2 or P5_096=-2 or P5_098=-2 or P5_100=-2 then pvmobile=-2;
if P5_083=-3 or P5_093=-3 or P5_096=-3 or P5_098=-3 or P5_100=-3 then pvmobile=-3;

/*A variable indicating the value of the property if the R lives in a house */
pvhouse=0;
if P5_101=1 and P5_112 not in (-1, -2, -3) then pvhouse=P5_112;
if P5_101=1 and (P5_112 eq -1 or P5_112 eq -2 or P5_112 eq -3) then do;
    if P5_113=1 then do; pvhouse=12500; flag=1; end;
    if P5_113=2 then do; pvhouse=37500; flag=1; end;
    if P5_113=3 then do; pvhouse=75000; flag=1; end;
    if P5_113=4 then do; pvhouse=175000; flag=1; end;
    if P5_113=5 then do; pvhouse=375000; flag=1; end;
    if P5_113=6 then do; pvhouse=750000; flag=1; end;
    if P5_113=7 then do; pvhouse=1000000; flag=1; end;
end;

if P5_101=-1 or P5_113=-1 then pvhouse=-1;
if P5_101=-2 or P5_113=-2 then pvhouse=-2;
if P5_101=-3 or P5_113=-3 then pvhouse=-3;

/*A variable indicating any mortgage or land contract, and second mortgages */
mortgage=0;
if (P5_114=1 or P5_114=2) and P5_115 not in (-1, -2, -3) then mortgage=P5_115;
if (P5_114=1 or P5_114=2) and (P5_115 eq -1 or P5_115 eq -2 or P5_115 eq -3) then do;
    if P5_116=1 then do; mortgage=12500; flag=1; end;
    if P5_116=2 then do; mortgage=37500; flag=1; end;
    if P5_116=3 then do; mortgage=75000; flag=1; end;
    if P5_116=4 then do; mortgage=175000; flag=1; end;
    if P5_116=5 then do; mortgage=375000; flag=1; end;
    if P5_116=6 then do; mortgage=750000; flag=1; end;
    if P5_116=7 then do; mortgage=1000001; flag=1; end;
end;

if P5_114=-1 or P5_116=-1 then mortgage=-1;
if P5_114=-2 or P5_116=-2 then mortgage=-2;
if P5_114=-3 or P5_116=-3 then mortgage=-3;

/* If a second mortgage...*/
secondmng=0;
if P5_117=1 and P5_118 not in (-1, -2, -3) then secondmng=P5_118;

```

```

if P5_117=1 and (P5_118 eq -1 or P5_118 eq -2 or P5_118 eq -3) then do;
  if P5_119=1 then do; secondmg=2500; flag=1; end;
  if P5_119=2 then do; secondmg=7500; flag=1; end;
  if P5_119=3 then do; secondmg=17500; flag=1; end;
  if P5_119=4 then do; secondmg=37500; flag=1; end;
  if P5_119=5 then do; secondmg=75000; flag=1; end;
  if P5_119=6 then do; secondmg=175000; flag=1; end;
  if P5_119=7 then do; secondmg=250001; flag=1; end;
  end;
if P5_117=-1 or P5_119=-1 then secondmg=-1;
if P5_117=-2 or P5_119=-2 then secondmg=-2;
if P5_117=-3 or P5_119=-3 then secondmg=-3;

```

```

/*A variable indicating whether the R and S own a business, partnership or professional practice */
ownbuss=0;
if P5_120=1 and P5_121 not in (-1, -2, -3) then ownbuss=P5_121;
if P5_120=1 and (P5_121 eq -1 or P5_121 eq -2 or P5_121 eq -3) then do;
  if P5_122=1 then do; ownbuss=12500; flag=1; end;
  if P5_122=2 then do; ownbuss=37500; flag=1; end;
  if P5_122=3 then do; ownbuss=75000; flag=1; end;
  if P5_122=4 then do; ownbuss=175000; flag=1; end;
  if P5_122=5 then do; ownbuss=375000; flag=1; end;
  if P5_122=6 then do; ownbuss=750000; flag=1; end;
  if P5_122=7 then do; ownbuss=1000001; flag=1; end;
  end;
if P5_120=-1 or P5_122=-1 then ownbuss=-1;
if P5_120=-2 or P5_122=-2 then ownbuss=-2;
if P5_120=-3 or P5_122=-3 then ownbuss=-3;

```

```

/*A variable indicating second homes, real estate, partnership, or money owed to R/S on land contract or mortgage */
realesta=0;
if P5_124=1 and P5_125 not in (-1, -2, -3) then realesta=P5_125;
if P5_124=1 and (P5_125 eq -1 or P5_125 eq -2 or P5_125 eq -3) then do;
  if P5_126=1 then do; realesta=12500; flag=1; end;
  if P5_126=2 then do; realesta=37500; flag=1; end;
  if P5_126=3 then do; realesta=75000; flag=1; end;
  if P5_126=4 then do; realesta=175000; flag=1; end;
  if P5_126=5 then do; realesta=375000; flag=1; end;
  if P5_126=6 then do; realesta=750000; flag=1; end;
  if P5_126=7 then do; realesta=1000001; flag=1; end;
  end;
if P5_124=-1 or P5_126=-1 then realesta=-1;
if P5_124=-2 or P5_126=-2 then realesta=-2;
if P5_124=-3 or P5_126=-3 then realesta=-3;

```

```

/*A variable indicating educational IRA accounts or other prepaid tuition savings accounts */
tuitions=0;
if P5_127=1 and P5_128 not in (-1, -2, -3) then tuitions=P5_128;
if P5_127=1 and (P5_128 eq -1 or P5_128 eq -2 or P5_128 eq -3) then do;
  if P5_129=1 then do; tuitions=2500; flag=1; end;
  if P5_129=2 then do; tuitions=7500; flag=1; end;
  if P5_129=3 then do; tuitions=17500; flag=1; end;
  if P5_129=4 then do; tuitions=37500; flag=1; end;
  if P5_129=5 then do; tuitions=75000; flag=1; end;
  if P5_129=6 then do; tuitions=175000; flag=1; end;
  if P5_129=7 then do; tuitions=250001; flag=1; end;
  end;

```

```

if P5_127=-1 or P5_129=-1 then tuitions=-1;
if P5_127=-2 or P5_129=-2 then tuitions=-2;
if P5_127=-3 or P5_129=-3 then tuitions=-3;

/*A variable indicating whether the R and S have any retirement or pension plans, such as 401K's */
retirepl=0;
if P5_130=1 and P5_131 not in (-1, -2, -3) then retirepl=P5_131;
if P5_130=1 and (P5_131 eq -1 or P5_131 eq -2 or P5_131 eq -3) then do;
    if P5_132=1 then do; retirepl=2500; flag=1; end;
    if P5_132=2 then do; retirepl=7500; flag=1; end;
    if P5_132=3 then do; retirepl=17500; flag=1; end;
    if P5_132=4 then do; retirepl=37500; flag=1; end;
    if P5_132=5 then do; retirepl=75000; flag=1; end;
    if P5_132=6 then do; retirepl=175000; flag=1; end;
    if P5_132=7 then do; retirepl=250001; flag=1; end;
end;
if P5_130=-1 or P5_132=-1 then retirepl=-1;
if P5_130=-2 or P5_132=-2 then retirepl=-2;
if P5_130=-3 or P5_132=-3 then retirepl=-3;

/*A variable indicating whether the R and S have any other savings in investment trusts, etc */
othsav1=0;
if P5_133=1 and P5_134 not in (-1, -2, -3) then othsav1=P5_134;
if P5_133=1 and (P5_134 eq -1 or P5_134 eq -2 or P5_134 eq -3) then do;
    if P5_135=1 then do; othsav1=2500; flag=1; end;
    if P5_135=2 then do; othsav1=7500; flag=1; end;
    if P5_135=3 then do; othsav1=17500; flag=1; end;
    if P5_135=4 then do; othsav1=37500; flag=1; end;
    if P5_135=5 then do; othsav1=75000; flag=1; end;
    if P5_135=6 then do; othsav1=175000; flag=1; end;
    if P5_135=7 then do; othsav1=250001; flag=1; end;
end;
if P5_133=-1 or P5_135=-1 then othsav1=-1;
if P5_133=-2 or P5_135=-2 then othsav1=-2;
if P5_133=-3 or P5_135=-3 then othsav1=-3;

/*A variable indicating whether the R and S have any other savings in CD's, Treasury bills, bonds, etc */
othsav2=0;
if P5_136=1 and P5_137 not in (-1, -2, -3) then othsav2=P5_137;
if P5_136=1 and (P5_137 eq -1 or P5_137 eq -2 or P5_137 eq -3) then do;
    if P5_138=1 then do; othsav2=2500; flag=1; end;
    if P5_138=2 then do; othsav2=7500; flag=1; end;
    if P5_138=3 then do; othsav2=17500; flag=1; end;
    if P5_138=4 then do; othsav2=37500; flag=1; end;
    if P5_138=5 then do; othsav2=75000; flag=1; end;
    if P5_138=6 then do; othsav2=175000; flag=1; end;
    if P5_138=7 then do; othsav2=250001; flag=1; end;
end;
if P5_136=-1 or P5_138=-1 then othsav2=-1;
if P5_136=-2 or P5_138=-2 then othsav2=-2;
if P5_136=-3 or P5_138=-3 then othsav2=-3;

/*A variable indicating whether the R and S have any other savings in mutual funds */
mutfunds=0;
if P5_139=1 and P5_140 not in (-1, -2, -3) then mutfunds=P5_140;
if P5_139=1 and (P5_140 eq -1 or P5_140 eq -2 or P5_140 eq -3) then do;
    if P5_141=1 then do; mutfunds=2500; flag=1; end;

```

```

if P5_141=2 then do; mutfunds=7500; flag=1; end;
if P5_141=3 then do; mutfunds=17500; flag=1; end;
if P5_141=4 then do; mutfunds=37500; flag=1; end;
if P5_141=5 then do; mutfunds=75000; flag=1; end;
if P5_141=6 then do; mutfunds=175000; flag=1; end;
if P5_141=7 then do; mutfunds=250001; flag=1; end;
end;

if P5_139=-1 or P5_141=-1 then mutfunds=-1;
if P5_139=-2 or P5_141=-2 then mutfunds=-2;
if P5_139=-3 or P5_141=-3 then mutfunds=-3;

/*A variable indicating the current market value of the R and S vehicles */
pvcars=0;
if P5_142=1 and P5_143 not in (-1, -2, -3) then pvcars=P5_143;
if P5_142=1 and (P5_143 eq -1 or P5_143 eq -2 or P5_143 eq -3) then do;
    if P5_144=1 then do; pvcars=500; flag=1; end;
    if P5_144=2 then do; pvcars=1750; flag=1; end;
    if P5_144=3 then do; pvcars=3750; flag=1; end;
    if P5_144=4 then do; pvcars=7500; flag=1; end;
    if P5_144=5 then do; pvcars=17500; flag=1; end;
    if P5_144=6 then do; pvcars=37500; flag=1; end;
    if P5_144=7 then do; pvcars=50001; flag=1; end;
end;
if P5_142=-1 or P5_144=-1 then pvcars=-1;
if P5_142=-2 or P5_144=-2 then pvcars=-2;
if P5_142=-3 or P5_144=-3 then pvcars=-3;

/*A variable indicating how much R and S owe on the vehicles */
owecar=0;
if P5_142=1 and P5_145 not in (-1, -2, -3) then owecar=P5_145;
if P5_142=1 and (P5_145 eq -1 or p5_145 eq -2 or p5_145 eq -3) then do;
    if P5_146=1 then do; owecar=500; flag=1; end;
    if P5_146=2 then do; owecar=1750; flag=1; end;
    if P5_146=3 then do; owecar=3750; flag=1; end;
    if P5_146=4 then do; owecar=7500; flag=1; end;
    if P5_146=5 then do; owecar=17500; flag=1; end;
    if P5_146=6 then do; owecar=37500; flag=1; end;
    if P5_146=7 then do; owecar=50001; flag=1; end;
end;
if P5_142=-1 or P5_146=-1 then owecar=-1;
if P5_142=-2 or P5_146=-2 then owecar=-2;
if P5_142=-3 or P5_146=-3 then owecar=-3;

/*A variable indicating the value of furniture */
pvfurnit=0;
if P5_147 ge 1 and P5_147=1 then do; pvfurnit=500; flag=1; end;
if P5_147 ge 1 and P5_147=2 then do; pvfurnit=1750; flag=1; end;
if P5_147 ge 1 and P5_147=3 then do; pvfurnit=3750; flag=1; end;
if P5_147 ge 1 and P5_147=4 then do; pvfurnit=7500; flag=1; end;
if P5_147 ge 1 and P5_147=5 then do; pvfurnit=17500; flag=1; end;
if P5_147 ge 1 and P5_147=6 then do; pvfurnit=37500; flag=1; end;
if P5_147 ge 1 and P5_147=7 then do; pvfurnit=50001; flag=1; end;
if P5_147=-1 then pvfurnit=-1;
if P5_147=-2 then pvfurnit=-2;
if P5_147=-3 then pvfurnit=-3;

/*A variable indicating any other assets owed to R and S by others */

```

```

othasset=0;
if P5_148=1 and P5_149 not in (-1, -2, -3) then othasset=P5_149;
if P5_148=1 and (P5_149 eq -1 or P5_149 eq -2 or P5_149 eq -3) then do;
    if P5_150=1 then do; othasset=2500; flag=1; end;
    if P5_150=2 then do; othasset=7500; flag=1; end;
    if P5_150=3 then do; othasset=17500; flag=1; end;
    if P5_150=4 then do; othasset=37500; flag=1; end;
    if P5_150=5 then do; othasset=75000; flag=1; end;
    if P5_150=6 then do; othasset=175000; flag=1; end;
    if P5_150=7 then do; othasset=250001; flag=1; end;
    end;
if P5_148=-1 or P5_150=-1 then othasset=-1;
if P5_148=-2 or P5_150=-2 then othasset=-2;
if P5_148=-3 or P5_150=-3 then othasset=-3;

/*A variable indicating any other debts R and S might have, such as:educational loans and any other debts */
eduloans=0;
if P5_151=1 and P5_152 not in (-1, -2, -3) then eduloans=P5_152;
if P5_151=1 and (P5_152 eq -1 or P5_152 eq -2 or P5_152 eq -3) then do;
    if P5_153=1 then do; eduloans=500; flag=1; end;
    if P5_153=2 then do; eduloans=1750; flag=1; end;
    if P5_153=3 then do; eduloans=3750; flag=1; end;
    if P5_153=4 then do; eduloans=7500; flag=1; end;
    if P5_153=5 then do; eduloans=17500; flag=1; end;
    if P5_153=6 then do; eduloans=37500; flag=1; end;
    if P5_153=7 then do; eduloans=50001; flag=1; end;
    end;
if P5_151=-1 or P5_153=-1 then eduloans=-1;
if P5_151=-2 or P5_153=-2 then eduloans=-2;
if P5_151=-3 or P5_153=-3 then eduloans=-3;

/* Any other debts...*/
othdebts=0;
if P5_154=1 and P5_155 not in (-1, -2, -3) then othdebts=P5_155;
if P5_154=1 and (P5_155 eq -1 or P5_155 eq -2 or P5_155 eq -3) then do;
    if P5_156=1 then do; othdebts=500; flag=1; end;
    if P5_156=2 then do; othdebts=1750; flag=1; end;
    if P5_156=3 then do; othdebts=3750; flag=1; end;
    if P5_156=4 then do; othdebts=7500; flag=1; end;
    if P5_156=5 then do; othdebts=17500; flag=1; end;
    if P5_156=6 then do; othdebts=37500; flag=1; end;
    if P5_156=7 then do; othdebts=50001; flag=1; end;
    end;
if P5_154=-1 or P5_156=-1 then othdebts=-1;
if P5_154=-2 or P5_156=-2 then othdebts=-2;
if P5_154=-3 or P5_156=-3 then othdebts=-3;

/* Calculate the household net worth according to the parent or: hhworthp=assets-liabilities */
hhworthp=-4;
hhworthp=(pvranrch + pvmobile + pvhouse + ownbuss + realesta + tuitions + retirepl + othsav1 + othsav2 + mutfunds
        + pvcars + pfurnit + othasset)-(mortgage + secondmg + owecar + eduloans + othdebts);
if pvranrch=-1 or pvmobile=-1 or pvhouse=-1 or ownbuss=-1 or realesta=-1 or tuitions=-1 or retirepl=-1 or othsav1=-1
    or othsav2=-1 or mutfunds=-1 or pvcars=-1 or pfurnit=-1 or othasset=-1 or mortgage=-1 or secondmg=-1
    or owecar=-1 or eduloans=-1 or othdebts=-1 then hhworthp=-1;
if pvranch=-2 or pvmobile=-2 or pvhouse=-2 or ownbuss=-2 or realesta=-2 or tuitions=-2 or retirepl=-2 or othsav1=-2
    or othsav2=-2 or mutfunds=-2 or pvcars=-2 or pfurnit=-2 or othasset=-2 or mortgage=-2 or secondmg=-2
    or owecar=-2 or eduloans=-2 or othdebts=-2 then hhworthp=-2;

```

Appendix 5: Income and Assets Variable Creation

```

if pvranch=-3 or pvmobile=-3 or pvhouse=-3 or ownbuss=-3 or realesta=-3 or tuitions=-3 or retirepl=-3 or othsav1=-3
or othsav2=-3 or mutfunds=-3 or pvcars=-3 or pfurnit=-3 or othasset=-3 or mortgage=-3 or secondmg=-3
or owecar=-3 or eduloans=-3 or othdebts=-3 then hhworthp=-3;
if pnocid=-4 then hhworthp=-4;

/* Create gross hh income according to the parent*/
groshhIp=-4;
do I=1 to 9;
if famI(I) ge 0 then do;
    groshhIp=(nfarmwgR + farmwgR+nfarmwgS + farmwgS + interest + AFDCADC + SSI + childsup + rentalI +
    famI01 + famI02 + famI03 + famI04 + famI05 + famI06 + famI07 + famI08 + famI09);
end;
end;
if nfarmwgR=-1 or farmwgR=-1 or nfarmwgS=-1 or farmwgS=-1 or interest=-1 or AFDCADC=-1 or SSI=-1 or
    childsup=-1 or rentalI=-1 or famI01=-1 or famI02=-1 or famI03=-1 or famI04=-1 or famI05=-1 or
    famI06=-1 or famI07=-1 or famI08=-1 or famI09=-1 then groshhIp=-1;
if nfarmwgR=-2 or farmwgR=-2 or nfarmwgS=-2 or farmwgS=-2 or interest=-2 or AFDCADC=-2 or SSI=-2 or
    childsup=-2 or rentalI=-2 or famI01=-2 or famI02=-2 or famI03=-2 or famI04=-2 or famI05=-2 or
    famI06=-2 or famI07=-2 or famI08=-2 or famI09=-2 then groshhIp=-2;
if nfarmwgR=-3 or farmwgR=-3 or nfarmwgS=-3 or farmwgS=-3 or interest=-3 or AFDCADC=-3 or SSI=-3 or
    childsup=-3 or rentalI=-3 or famI01=-3 or famI02=-3 or famI03=-3 or famI04=-3 or famI05=-3 or
    famI06=-3 or famI07=-3 or famI08=-3 or famI09=-3 then groshhIp=-3;
if pnocid=-4 then groshhIp=-4;

```

***** SECTION 2: YOUTH HOUSEHOLD NET WORTH AND GROSS HOUSEHOLD INCOME *****

flag=0;

```

/*First, create a variable indicating net receipts from non-farm employment earned by youth (Y), such as wages
(nfarmwgY). */
nfarmwgY=0;
if (YI_1400=1 and YI_1700 not in (-1, -2, -3)) or (YI_1400=-1 and YI_1600=1 and YI_1700 not in (-1, -2, -3)) or
(YI_1400=-2 and YI_1500=1 and YI_1700 not in (-1, -2, -3)) or (YI_1400=-2 and YI_1500=-1 and
YI_1600=1 and YI_1700 not in (-1, -2, -3)) then nfarmwgY=YI_1700;
if (YI_1400=1 or (YI_1400=-1 and YI_1600=1) or (YI_1400=-2 and YI_1500=1) or (YI_1400=-2 and YI_1500=-1
and YI_1600=1) or (YI_1400=-3 and YI_1600=1) or (YI_1400=-3 and YI_1500=1) or (YI_1400=-2 and YI_1500=-3
and YI_1600=1)) and (YI_1700 eq -1 or YI_1700 eq -2 or YI_1700 eq -3) then do;
    if YI_1800=1 then do; nfarmwgY=2500; flag=1; end;
    if YI_1800=2 then do; nfarmwgY=7500; flag=1; end;
    if YI_1800=3 then do; nfarmwgY=17500; flag=1; end;
    if YI_1800=4 then do; nfarmwgY=22500; flag=1; end;
    if YI_1800=5 then do; nfarmwgY=75000; flag=1; end;
    if YI_1800=6 then do; nfarmwgY=175000; flag=1; end;
    if YI_1800=7 then do; nfarmwgY=250001; flag=1; end;
end;
if YI_1600=-1 or YI_1800=-1 then nfarmwgY=-1;
if YI_1500=-2 or YI_1800=-2 then nfarmwgY=-2;
if YI_1600=-3 or YI_1500=-3 or YI_1800=-3 then nfarmwgY=-3;

```

/* For all the questions below, the youth must be INDEPENDENT (YI_1900=1) */

```

/*Second, create a variable indicating net receipts from farm self-employment earned by the Y (farmwgY)*/
farmwgY=0;
if YI_1900=1 and YI_2000=1 and YI_2100 not in (-1, -2, -3) then farmwgY=YI_2100;
if YI_1900=1 and YI_2000=1 and (YI_2100 eq -1 or YI_2100 eq -2 or YI_2100 eq -3) then do;
    if YI_2200=1 then do; farmwgY=-2; flag=1; end;

```

```

if YI_2200=2 then do; farmwgY=2500; flag=1; end;
if YI_2200=3 then do; farmwgY=7500; flag=1; end;
if YI_2200=4 then do; farmwgY=17500; flag=1; end;
if YI_2200=5 then do; farmwgY=37500; flag=1; end;
if YI_2200=6 then do; farmwgY=75000; flag=1; end;
if YI_2200=7 then do; farmwgY=175000; flag=1; end;
if YI_2200=8 then do; farmwgY=250001; flag=1; end;
end;

if YI_2000=-1 or YI_2200=-1 then farmwgY=-1;
if YI_2000=-2 or YI_2200=-2 then farmwgY=-2;
if YI_2000=-3 or YI_2200=-3 then farmwgY=-3;

/*Third, create the above variables for the event the Y has a spouse/partner: nfarmwgP and farmwgP */
nfarmwgP=0;
if YI_1900=1 then do;
if (YI_2300=1 and YI_2400=1 and YI_2600 not in (-1, -2, -3)) or (YI_2300=1 and YI_2400=-1 and YI_2500=1 and
    YI_2600 not in (-1, -2, -3)) then nfarmwgP=YI_2600;
if (YI_2300=1 and YI_2400=1 and (YI_2600 eq -1 or YI_2600 eq -2 or YI_2600 eq -3)) or (YI_2300=1 and
    YI_2400=-1 and YI_2500=1 and (YI_2600 eq -1 or YI_2600 eq -2 or YI_2600 eq -3)) then do;
    if YI_2700=1 then do; nfarmwgP=2500; flag=1; end;
    if YI_2700=2 then do; nfarmwgP=7500; flag=1; end;
    if YI_2700=3 then do; nfarmwgP=17500; flag=1; end;
    if YI_2700=4 then do; nfarmwgP=37500; flag=1; end;
    if YI_2700=5 then do; nfarmwgP=75000; flag=1; end;
    if YI_2700=6 then do; nfarmwgP=175000; flag=1; end;
    if YI_2700=7 then do; nfarmwgP=250001; flag=1; end;
end;

if YI_2300=-1 or YI_2500=-1 or YI_2700=-1 then nfarmwgP=-1;
if YI_2300=-2 or YI_2400=-2 or YI_2700=-2 then nfarmwgP=-2;
if YI_2300=-3 or YI_2400=-3 or YI_2500=-3 or YI_2700=-3 then nfarmwgP=-3;
end;

farmwgP=0;
if YI_1900=1 then do;
if YI_2900=1 and YI_3000 not in (-1, -2, -3) then farmwgP=YI_3000;
if YI_2900=1 and (YI_3000 eq -1 or YI_3000 eq -2 or YI_3000 eq -3) then do;
    if YI_3100=1 then do; farmwgP=-2; flag=1; end;
    if YI_3100=2 then do; farmwgP=2500; flag=1; end;
    if YI_3100=3 then do; farmwgP=7500; flag=1; end;
    if YI_3100=4 then do; farmwgP=17500; flag=1; end;
    if YI_3100=5 then do; farmwgP=37500; flag=1; end;
    if YI_3100=6 then do; farmwgP=75000; flag=1; end;
    if YI_3100=7 then do; farmwgP=175000; flag=1; end;
    if YI_3100=8 then do; farmwgP=250001; flag=1; end;
end;

if YI_2300=-1 or YI_2900=-1 or YI_3100=-1 then farmwgP=-1;
if YI_2300=-2 or YI_2900=-2 or YI_3100=-2 then farmwgP=-2;
if YI_2300=-3 or YI_2900=-3 or YI_3100=-3 then farmwgP=-3;
end;

/* Create a variable indicating whether they collected any child support (childsuY) */
childsuY=0;
if YI_1900=1 then do;
if YI_4000=1 and YI_4100 not in (-1, -2, -3) then childsuY=YI_4100;
if YI_4000=1 and (YI_4100 eq -1 or YI_4100 eq -2 or YI_4100 eq -3) then do;
    if YI_4200=1 then do; childsuY=500; flag=1; end;
    if YI_4200=2 then do; childsuY=1750; flag=1; end;

```

```

if YI_4200=3 then do; childsuY=3750; flag=1; end;
if YI_4200=4 then do; childsuY=7500; flag=1; end;
if YI_4200=5 then do; childsuY=17500; flag=1; end;
if YI_4200=6 then do; childsuY=37500; flag=1; end;
if YI_4200=7 then do; childsuY=50001; flag=1; end;
end;

if YI_4000=-1 or YI_4200=-1 then childsuY=-1;
if YI_4000=-2 or YI_4200=-2 then childsuY=-2;
if YI_4000=-3 or YI_4200=-3 then childsuY=-3;
end;

/* Create a variable indicating the amount of interests received by the youth (Y) and partner/spouse */
interesY=0;
if YI_1900=1 then do;
if YI_4300=1 and YI_4400 not in (-1, -2, -3) then interesY=YI_4400;
if YI_4300=1 and (YI_4400 eq -1 or YI_4400 eq -2 or YI_4400 eq -3) then do;
    if YI_4500=1 then do; interesY=250; flag=1; end;
    if YI_4500=2 then do; interesY=750; flag=1; end;
    if YI_4500=3 then do; interesY=1750; flag=1; end;
    if YI_4500=4 then do; interesY=3750; flag=1; end;
    if YI_4500=5 then do; interesY=6250; flag=1; end;
    if YI_4500=6 then do; interesY=8750; flag=1; end;
    if YI_4500=7 then do; interesY=10001; flag=1; end;
end;
if YI_4300=-1 or YI_4500=-1 then interesY=-1;
if YI_4300=-2 or YI_4500=-2 then interesY=-2;
if YI_4300=-3 or YI_4500=-3 then interesY=-3;
end;

/* Create a variable indicating whether they collected any dividends from stocks and mutual funds */
dividend=0;
if YI_1900=1 then do;
if YI_4600=1 and YI_4700 not in (-1, -2, -3) then dividend=YI_4700;
if YI_4600=1 and (YI_4700 eq -1 or YI_4700 eq -2 or YI_4700 eq -3) then do;
    if YI_4800=1 then do; dividend=250; flag=1; end;
    if YI_4800=2 then do; dividend=750; flag=1; end;
    if YI_4800=3 then do; dividend=1750; flag=1; end;
    if YI_4800=4 then do; dividend=3750; flag=1; end;
    if YI_4800=5 then do; dividend=6250; flag=1; end;
    if YI_4800=6 then do; dividend=8750; flag=1; end;
    if YI_4800=7 then do; dividend=10001; flag=1; end;
end;
if YI_4600=-1 or YI_4800=-1 then dividend=-1;
if YI_4600=-2 or YI_4800=-2 then dividend=-2;
if YI_4600=-3 or YI_4800=-3 then dividend=-3;
end;

/* Create a variable indicating any rental income */
rentalIY=0;
if YI_1900=1 then do;
if YI_4900=1 and YI_5000 not in (-1, -2, -3) then rentalIY=YI_5000;
if YI_4900=1 and (YI_5000 eq -1 or YI_5000 eq -2 or YI_5000 eq -3) then do;
    if YI_5100=1 then do; rentalIY=500; flag=1; end;
    if YI_5100=2 then do; rentalIY=1750; flag=1; end;
    if YI_5100=3 then do; rentalIY=3750; flag=1; end;
    if YI_5100=4 then do; rentalIY=7500; flag=1; end;
    if YI_5100=5 then do; rentalIY=17500; flag=1; end;
end;

```

```

if YI_5100=6 then do; rentalIY=37500; flag=1; end;
if YI_5100=7 then do; rentalIY=50001; flag=1; end;
end;
if YI_4900=-1 or YI_5100=-1 then rentalIY=-1;
if YI_4900=-2 or YI_5100=-2 then rentalIY=-2;
if YI_4900=-3 or YI_5100=-3 then rentalIY=-3;
end;

/* next a variable indicating whether they received any property/money from estates, trusts, annuities, inheritances */
estatesY=0;
if YI_1900=1 then do;
if YI_5200=1 and YI_5300 not in (-1, -2, -3) then estatesY=YI_5300;
if YI_5200=1 and (YI_5300 eq -1 or YI_5300 eq -2 or YI_5300 eq -3) then do;
    if YI_5400=1 then do; estatesY=2500; flag=1; end;
    if YI_5400=2 then do; estatesY=7500; flag=1; end;
    if YI_5400=3 then do; estatesY=17500; flag=1; end;
    if YI_5400=4 then do; estatesY=37500; flag=1; end;
    if YI_5400=5 then do; estatesY=75000; flag=1; end;
    if YI_5400=6 then do; estatesY=175000; flag=1; end;
    if YI_5400=7 then do; estatesY=250001; flag=1; end;
end;
if YI_5200=-1 or YI_5400=-1 then estatesY=-1;
if YI_5200=-2 or YI_5400=-2 then estatesY=-2;
if YI_5200=-3 or YI_5400=-3 then estatesY=-3;
end;

/* Now create a variable indicating any cash or money other than allowances parents might have given the youth if
he/she still lives with them (INCLUDED IN THE INCOME) */
allowpar=0;
if YI_1900=1 and YI_5600=1 then do;
if YI_5700=1 and YI_5800 not in (-1, -2, -3) then allowpar=YI_5800;
if YI_5700=1 and (YI_5800 eq -1 or YI_5800 eq -2 or YI_5800 eq -3) then do;
    if YI_5900=1 then do; allowpar=250; flag=1; end;
    if YI_5900=2 then do; allowpar=750; flag=1; end;
    if YI_5900=3 then do; allowpar=1750; flag=1; end;
    if YI_5900=4 then do; allowpar=3750; flag=1; end;
    if YI_5900=5 then do; allowpar=6250; flag=1; end;
    if YI_5900=6 then do; allowpar=8750; flag=1; end;
    if YI_5900=7 then do; allowpar=10001; flag=1; end;
end;
if YI_1900=0 or YI_5700=-4 then allowpar=-4;
if YI_5700=-1 or YI_5900=-1 then allowpar=-1;
if YI_5700=-2 or YI_5900=-2 then allowpar=-2;
if YI_5700=-3 or YI_5900=-3 then allowpar=-3;
end;

/* If the youth lived with the parents during the previous year and paid for room and board...*/
roomyear=0;
if YI_1900=1 and YI_5600=1 then do;
if YI_6100=1 and YI_6200 not in (-1, -2, -3) then do;
    if YI_6300=1 then roomyear=YI_6200*52;
    if YI_6300=2 then roomyear=YI_6200*26;
    if YI_6300=3 then roomyear=YI_6200*12;
    if YI_6300=4 then roomyear=YI_6200*2;
    if YI_6300=5 then roomyear=YI_6200;
    if YI_6300=6 then do; roomyear=YI_6200; flag=1; end;
    if YI_6300=7 then do; roomyear=YI_6200; flag=1; end;

```

```

        end;
if YI_6100=-1 or YI_6200=-1 or YI_6300=-1 then roomyear=-1;
if YI_6100=-2 or YI_6200=-2 or YI_6300=-2 then roomyear=-2;
if YI_6100=-3 or YI_6200=-3 or YI_6300=-3 then roomyear=-3;
end;

/* Any other payments to your parents?, if the youth lived with them during the previous year */
othpaypa=0;
if YI_5600=1 then do;
if YI_1900=1 and YI_6100=0 and YI_6310=1 then othpaypa=YI_6320;
if YI_1900=0 or YI_6310=1 or YI_6320=-1 then othpaypa=-4;
if YI_6310=-1 or YI_6320=-1 then othpaypa=-1;
if YI_6310=-2 or YI_6320=-2 then othpaypa=-2;
if YI_6310=-3 or YI_6320=-3 then othpaypa=-3;
end;

/* If living with mother/female guardian, did R receive any money from her? (incl. in youth income definition) */
allowmot=0;
if YI_1900=1 and YI_6400=1 then do;
if YI_6500=1 and YI_6600 not in (-1, -2, -3) then allowmot=YI_6600;
if YI_6500=1 and (YI_6600 eq -1 or YI_6600 eq -2 or YI_6600 eq -3) then do;
    if YI_6700=1 then do; allowmot=250; flag=1; end;
    if YI_6700=2 then do; allowmot=750; flag=1; end;
    if YI_6700=3 then do; allowmot=1750; flag=1; end;
    if YI_6700=4 then do; allowmot=3750; flag=1; end;
    if YI_6700=5 then do; allowmot=7500; flag=1; end;
    if YI_6700=6 then do; allowmot=8750; flag=1; end;
    if YI_6700=7 then do; allowmot=10001; flag=1; end;
end;
if YI_6500=-1 or YI_6700=-1 then allowmot=-1;
if YI_6500=-2 or YI_6700=-2 then allowmot=-2;
if YI_6500=-3 or YI_6700=-3 then allowmot=-3;
end;

/* If living with the mother or female guardian but rather made payments to this person...*/
othpaymo=0;
if YI_6400=1 then do;
if YI_1900=1 and YI_6800=1 then othpaymo=YI_6900;
if YI_6800=-1 or YI_6900=-1 then othpaymo=-1;
if YI_6800=-2 or YI_6900=-2 then othpaymo=-2;
if YI_6800=-3 or YI_6900=-3 then othpaymo=-3;
end;

/* If lived with the mother or female guardian and made room and board payments to her...*/
yroommot=0;
if YI_1900=1 and YI_6400=1 then do;
if YI_6910=1 and YI_6920=1 and YI_6940=1 then yroommot=YI_6930*52;
if YI_6910=1 and YI_6920=1 and YI_6940=2 then yroommot=YI_6930*26;
if YI_6910=1 and YI_6920=1 and YI_6940=3 then yroommot=YI_6930*12;
if YI_6910=1 and YI_6920=1 and YI_6940=4 then yroommot=YI_6930*2;
if YI_6910=1 and YI_6920=1 and YI_6940=5 then yroommot=YI_6930;
if YI_6910=1 and YI_6920=1 and YI_6940=6 then do; yroommot=YI_6930; flag=1; end;
if YI_6910=1 and YI_6920=1 and YI_6940=7 then do; yroommot=YI_6930; flag=1; end;
if YI_6910=-1 or YI_6920=-1 or YI_6930=-1 or YI_6940=-1 then yroommot=-1;
if YI_6910=-2 or YI_6920=-2 or YI_6930=-2 or YI_6940=-2 then yroommot=-2;
if YI_6910=-3 or YI_6920=-3 or YI_6930=-3 or YI_6940=-3 then yroommot=-3;
end;

```

```

/* If the youth lived with the father or male guardian and received any cash or money other than allowances from
him (INCLUDED IN THE YOUTH INCOME) */
allowfat=0;
if YI_1900=1 then do;
if YI_7000=1 and YI_7100=1 and YI_7200 not in (-1, -2, -3) then allowfat=YI_7200;
if YI_7000=1 and YI_7100=1 and (YI_7200 eq -1 or YI_7200 eq -2 or YI_7200 eq -3) then do;
    if YI_7300=1 then do; allowfat=250; flag=1; end;
    if YI_7300=2 then do; allowfat=750; flag=1; end;
    if YI_7300=3 then do; allowfat=1750; flag=1; end;
    if YI_7300=4 then do; allowfat=3750; flag=1; end;
    if YI_7300=5 then do; allowfat=6250; flag=1; end;
    if YI_7300=6 then do; allowfat=8750; flag=1; end;
    if YI_7300=7 then do; allowfat=10001; flag=1; end;
end;
if YI_7100=-1 or YI_7300=-1 then allowfat=-1;
if YI_7100=-2 or YI_7300=-2 then allowfat=-2;
if YI_7100=-3 or YI_7300=-3 then allowfat=-3;
end;

/* If the youth lived with the father or male guardian but rather paid him...*/
othpayfa=0;
if YI_1900=1 and YI_7000=1 and YI_7400=1 then othpayfa=YI_7500;
if YI_7400=-1 or YI_7500=-1 then othpayfa=-1;
if YI_7400=-2 or YI_7500=-2 then othpayfa=-2;
if YI_7400=-3 or YI_7500=-3 then othpayfa=-3;

/* If the youth lived with the father or male guardian and paid him for room and board */
yroomfat=0;
if YI_1900=1 and YI_7000=1 then do;
if YI_7520=1 and YI_7540=1 then yroomfat=YI_7530*52;
if YI_7520=1 and YI_7540=2 then yroomfat=YI_7530*26;
if YI_7520=1 and YI_7540=3 then yroomfat=YI_7530*12;
if YI_7520=1 and YI_7540=4 then yroomfat=YI_7530*2;
if YI_7520=1 and YI_7540=5 then yroomfat=YI_7530;
if YI_7520=1 and YI_7540=6 then do; yroomfat=YI_7530; flag=1; end;
if YI_7520=1 and YI_7540=7 then do; yroomfat=YI_7530; flag=1; end;
if YI_7520=-1 or YI_7540=-1 then yroomfat=-1;
if YI_7520=-2 or YI_7540=-2 then yroomfat=-2;
if YI_7520=-3 or YI_7540=-3 then yroomfat=-3;
end;

/* Income received by the youth from other sources: SS payments, pension and retirement income, alimony,
payments from insurance policies, etc...? */
pensionY=0;
if YI_1900=1 then do;
if YI_7600=1 and YI_7700 not in (-1, -2, -3) then pensionY=YI_7700;
if YI_7600=1 and (YI_7700 eq -1 or YI_7700 eq -2 or YI_7700 eq -3) then do;
    if YI_7800=1 then do; pensionY=500; flag=1; end;
    if YI_7800=2 then do; pensionY=1750; flag=1; end;
    if YI_7800=3 then do; pensionY=3750; flag=1; end;
    if YI_7800=4 then do; pensionY=7500; flag=1; end;
    if YI_7800=5 then do; pensionY=17500; flag=1; end;
    if YI_7800=6 then do; pensionY=37500; flag=1; end;
    if YI_7800=7 then do; pensionY=50001; flag=1; end;
end;
if YI_7600=-1 or YI_7800=-1 then pensionY=-1;

```

```

if YI_7600=-2 or YI_7800=-2 then pensionY=-2;
if YI_7600=-3 or YI_7800=-3 then pensionY=-3;
end;

/* For EVERYONE, DEPENDENT OR INDEPENDENT, allowances received by the youth from his/her family
(NOT INCLUDED IN THE INCOME DEFINITION OF THE YOUTH) */
yfaallow=0;
if YI_8100=1 and YI_8200 not in (0, -1, -2, -3) then do;
    if YI_8300=1 then yfaallow=YI_8200*52;
    if YI_8300=2 then yfaallow=YI_8200*12;
    if YI_8300=3 then do; yfaallow=YI_8200; flag=1; end;
    end;
if YI_8100=-1 or YI_8200=-1 or YI_8300=-1 then yfaallow=-1;
if YI_8100=-2 or YI_8200=-2 or YI_8300=-2 then yfaallow=-2;
if YI_8100=-3 or YI_8200=-3 or YI_8300=-3 then yfaallow=-3;

/* Now a few more questions regarding income if the individual is INDEPENDENT (YI_8500=1) OR 14 YEARS
OF AGE OR OLDER (YI_8400=1) */

/* If the youth lived with the father, the father's income */
faincome=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_8600=1 and YI_8700=1 and YI_8800 not in (-1, -2, -3) then faincome=YI_8800;
if YI_8600=1 and YI_8700=1 and (YI_8800 eq -1 or YI_8800 eq -2 or YI_8800 eq -3) then do;
    if YI_8900=1 then do; faincome=2500; flag=1; end;
    if YI_8900=2 then do; faincome=7500; flag=1; end;
    if YI_8900=3 then do; faincome=17500; flag=1; end;
    if YI_8900=4 then do; faincome=37500; flag=1; end;
    if YI_8900=5 then do; faincome=75000; flag=1; end;
    if YI_8900=6 then do; faincome=175000; flag=1; end;
    if YI_8900=7 then do; faincome=250001; flag=1; end;
    end;
if YI_8600=-1 or YI_8700=-1 or YI_8900=-1 then faincome=-1;
if YI_8600=-2 or YI_8700=-2 or YI_8900=-2 then faincome=-2;
if YI_8600=-3 or YI_8700=-3 or YI_8900=-3 then faincome=-3;
end;

/* If the youth lived with the mother, the mother's income */
maincome=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_9100=1 and YI_9200=1 and YI_9300 not in (-1, -2, -3) then maincome=YI_9300;
if YI_9100=1 and YI_9200=1 and (YI_9300 eq -1 or YI_9300 eq -2 or YI_9300 eq -3) then do;
    if YI_9400=1 then do; maincome=2500; flag=1; end;
    if YI_9400=2 then do; maincome=7500; flag=1; end;
    if YI_9400=3 then do; maincome=17500; flag=1; end;
    if YI_9400=4 then do; maincome=37500; flag=1; end;
    if YI_9400=5 then do; maincome=75000; flag=1; end;
    if YI_9400=6 then do; maincome=175000; flag=1; end;
    if YI_9400=7 then do; maincome=250001; flag=1; end;
    end;
if YI_9100=-1 or YI_9200=-1 or YI_9400=-1 then maincome=-1;
if YI_9100=-2 or YI_9200=-2 or YI_9400=-2 then maincome=-2;
if YI_9100=-3 or YI_9200=-3 or YI_9400=-3 then maincome=-3;
end;

/* If the youth lives with the male guardian */
mgincome=0;

```

```

if YI_8400=1 or YI_8500=1 then do;
if YI_9600=1 and YI_9700=1 and YI_9800 not in (-1, -2, -3) then mgincome=YI_9800;
if YI_9600=1 and YI_9700=1 and (YI_9800 eq -1 or YI_9800 eq -2 or YI_9800 eq -3) then do;
    if YI_9900=1 then do; mgincome=2500; flag=1; end;
    if YI_9900=2 then do; mgincome=7500; flag=1; end;
    if YI_9900=3 then do; mgincome=17500; flag=1; end;
    if YI_9900=4 then do; mgincome=37500; flag=1; end;
    if YI_9900=5 then do; mgincome=75000; flag=1; end;
    if YI_9900=6 then do; mgincome=175000; flag=1; end;
    if YI_9900=7 then do; mgincome=250001; flag=1; end;
    end;
if YI_9600=-1 or YI_9700=-1 or YI_9900=-1 then mgincome=-1;
if YI_9600=-2 or YI_9700=-2 or YI_9900=-2 then mgincome=-2;
if YI_9600=-3 or YI_9700=-3 or YI_9900=-3 then mgincome=-3;
end;

/* If the youth lives with female guardian */
fgincome=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_10100=1 and YI_10200=1 and YI_10300 not in (-1, -2, -3) then fgincome=YI_10300;
if YI_10100=1 and YI_10200=1 and (YI_10300 eq -1 or YI_10300 eq -2 or YI_10300 eq -3) then do;
    if YI_10400=1 then do; fgincome=2500; flag=1; end;
    if YI_10400=2 then do; fgincome=7500; flag=1; end;
    if YI_10400=3 then do; fgincome=17500; flag=1; end;
    if YI_10400=4 then do; fgincome=37500; flag=1; end;
    if YI_10400=5 then do; fgincome=75000; flag=1; end;
    if YI_10400=6 then do; fgincome=175000; flag=1; end;
    if YI_10400=7 then do; fgincome=250001; flag=1; end;
    end;
if YI_10100=-1 or YI_10200=-1 or YI_10400=-1 then fgincome=-1;
if YI_10100=-2 or YI_10200=-2 or YI_10400=-2 then fgincome=-2;
if YI_10100=-3 or YI_10200=-3 or YI_10400=-3 then fgincome=-3;
end;

/* Check for income from any household member 14 years of age or older and who hasn't been asked before */
array otfamI otfamI01 otfamI02 otfamI03 otfamI04 otfamI05 otfamI06 otfamI07 otfamI08 otfamI09;
array I11100 I111001 I111002 I111003 I111004 I111005 I111006 I111007 I111008 I111009 ;
array I11600 I116001 I116002 I116003 I111004 I116005 I116006 I116007 I116008 I116009 ;
array I11700 I117001 I117002 I117003 I117004 I117005 I117006 I117007 I117008 I117009 ;

do I=1 to 9;
otfamI(I)=0;
if (YI_8400=1 or YI_8500=1) and YI_10700=1 then do;
    if I11100(I)=0 and I11600(I) not in (-1, -2, -3, -4) then otfamI(I)=I11600(I);
    if I11100(I)=0 and (I11600(I) eq -1 or I11600(I) eq -2 or I11600(I) eq -3) then do;
        if I11700(I)=1 then do; otfamI(I)=2500; flag=1; end;
        if I11700(I)=2 then do; otfamI(I)=7500; flag=1; end;
        if I11700(I)=3 then do; otfamI(I)=17500; flag=1; end;
        if I11700(I)=4 then do; otfamI(I)=37500; flag=1; end;
        if I11700(I)=5 then do; otfamI(I)=75000; flag=1; end;
        if I11700(I)=6 then do; otfamI(I)=175000; flag=1; end;
        if I11700(I)=7 then do; otfamI(I)=250001; flag=1; end;
        end;
    if I11100(I)=-1 or I11700(I)=-1 then otfamI(I)=-1;
    if I11100(I)=-2 or I11700(I)=-2 then otfamI(I)=-2;
    if I11100(I)=-3 or I11700(I)=-3 then otfamI(I)=-3;
    end;

```

end;

```

/* If the youth lives on ranch or farm and owns it, the value of the ranch */
pvranchY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_12200=1 and YI_12400=1 and YI_12600 not in (-1, -2, -3) then pvranchY=YI_12600;
if YI_12200=1 and YI_12400=1 and (YI_12600 eq -1 or YI_12600 eq -2 or YI_12600 eq -3) then do;
    if YI_12700=1 then do; pvranchY=12500; flag=1; end;
    if YI_12700=2 then do; pvranchY=37500; flag=1; end;
    if YI_12700=3 then do; pvranchY=75000; flag=1; end;
    if YI_12700=4 then do; pvranchY=175000; flag=1; end;
    if YI_12700=5 then do; pvranchY=375000; flag=1; end;
    if YI_12700=6 then do; pvranchY=750000; flag=1; end;
    if YI_12700=7 then do; pvranchY=1000001; flag=1; end;
    end;
if YI_12200=1 and YI_12400=2 and YI_12800 ge 1 and YI_12900 not in (-1, -2, -3) then pvranchY=YI_12900;
if YI_12200=1 and YI_12400=2 and YI_12800 ge 1 and (YI_12900= -1 or YI_12900= -2 or YI_12900= -3) then do;
    if YI_13000=1 then do; pvranchY=12500; flag=1; end;
    if YI_13000=2 then do; pvranchY=37500; flag=1; end;
    if YI_13000=3 then do; pvranchY=75000; flag=1; end;
    if YI_13000=4 then do; pvranchY=175000; flag=1; end;
    if YI_13000=5 then do; pvranchY=375000; flag=1; end;
    if YI_13000=6 then do; pvranchY=750000; flag=1; end;
    if YI_13000=7 then do; pvranchY=1000001; flag=1; end;
    end;
if YI_12200=1 and YI_12400=2 and (YI_12800 eq -1 or YI_12800 eq -2 or YI_12800 eq -3) and YI_13100 not in
(-1, -2, -3) then pvranchY=YI_13100;
if YI_12200=1 and YI_12400=2 and (YI_12800 eq -1 or YI_12800 eq -2 or YI_12800 eq -3) and (YI_13100 eq -1
or YI_13100 eq -2 or YI_13100 eq -3) then do;
    if YI_13200=1 then do; pvranchY=12500; flag=1; end;
    if YI_13200=2 then do; pvranchY=37500; flag=1; end;
    if YI_13200=3 then do; pvranchY=75000; flag=1; end;
    if YI_13200=4 then do; pvranchY=175000; flag=1; end;
    if YI_13200=5 then do; pvranchY=375000; flag=1; end;
    if YI_13200=6 then do; pvranchY=750000; flag=1; end;
    if YI_13200=7 then do; pvranchY=1000001; flag=1; end;
    end;
if YI_12200=-1 | YI_12400=-1 | YI_12700=-1 | YI_12800=-1 | YI_13000=-1 | YI_13100=-1 then pvranch=-1;
if YI_12200=-2 | YI_12400=-2 | YI_12700=-2 | YI_12800=-2 | YI_13000=-2 | YI_13100=-2 then pvranch=-2;
if YI_12200=-3 | YI_12400=-3 | YI_12700=-3 | YI_12800=-3 | YI_13000=-3 | YI_13100=-3 then pvranch=-3;
end;

/* If the youth lives in a mobile home and owns it, the value of the mobile home */
pvmobilY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_12300=1 and YI_13300=1 and YI_13500 not in (-1, -2, -3) then pvmobilY=YI_13500;
if YI_12300=1 and YI_13300=1 and (YI_13500 eq -1 or YI_13500 eq -2 or YI_13500 eq -3) then do;
    if YI_13600=1 then do; pvmobilY=12500; flag=1; end;
    if YI_13600=2 then do; pvmobilY=37500; flag=1; end;
    if YI_13600=3 then do; pvmobilY=75000; flag=1; end;
    if YI_13600=4 then do; pvmobilY=175000; flag=1; end;
    if YI_13600=5 then do; pvmobilY=375000; flag=1; end;
    if YI_13600=6 then do; pvmobilY=750000; flag=1; end;
    if YI_13600=7 then do; pvmobilY=1000001; flag=1; end;
    end;
if YI_12300=1 and YI_13300=2 and YI_13700 not in (-1, -2, -3) then pvmobilY=YI_13700;
if YI_12300=1 and YI_13300=2 and (YI_13700 eq -1 or YI_13700 eq -2 or YI_13700 eq -3) then do;

```

```

if YI_13800=1 then do; pvmobilY=12500; flag=1; end;
if YI_13800=2 then do; pvmobilY=37500; flag=1; end;
if YI_13800=3 then do; pvmobilY=75000; flag=1; end;
if YI_13800=4 then do; pvmobilY=175000; flag=1; end;
if YI_13800=5 then do; pvmobilY=375000; flag=1; end;
if YI_13800=6 then do; pvmobilY=750000; flag=1; end;
if YI_13800=7 then do; pvmobilY=1000001; flag=1; end;
end;

if YI_12300=1 and YI_13300=3 and YI_14200 not in (-1, -2, -3) then pvmobilY=YI_14200;
if YI_12300=1 and YI_13300=3 and (YI_14200 eq -1 or YI_14200 eq -2 or YI_14200 eq -3) then do;
    if YI_14300=1 then do; pvmobilY=12500; flag=1; end;
    if YI_14300=2 then do; pvmobilY=37500; flag=1; end;
    if YI_14300=3 then do; pvmobilY=75000; flag=1; end;
    if YI_14300=4 then do; pvmobilY=175000; flag=1; end;
    if YI_14300=5 then do; pvmobilY=375000; flag=1; end;
    if YI_14300=6 then do; pvmobilY=750000; flag=1; end;
    if YI_14300=7 then do; pvmobilY=1000001; flag=1; end;
end;

if YI_12300=-1 | YI_12300=-2 | YI_13300=-1 | YI_13600=-1 | YI_13800=-1 | YI_14300=-1 then pvmobilY=-1;
if YI_12300=-2 | YI_12300=-2 | YI_13300=-2 | YI_13600=-2 | YI_13800=-2 | YI_14300=-2 then pvmobilY=-2;
if YI_12300=-3 | YI_12300=-3 | YI_13300=-3 | YI_13600=-3 | YI_13800=-3 | YI_14300=-3 then pvmobilY=-3;
end;

/* If the youth owns an apartment or house, its present value */
pvhomedY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_14700=1 and YI_15800 not in (-1, -2, -3) then pvhomedY=YI_15800;
if YI_14700=1 and (YI_15800 eq -1 or YI_15800 eq -2 or YI_15800 eq -3) then do;
    if YI_15900=1 then do; pvhomedY=12500; flag=1; end;
    if YI_15900=2 then do; pvhomedY=37500; flag=1; end;
    if YI_15900=3 then do; pvhomedY=75000; flag=1; end;
    if YI_15900=4 then do; pvhomedY=175000; flag=1; end;
    if YI_15900=5 then do; pvhomedY=375000; flag=1; end;
    if YI_15900=6 then do; pvhomedY=750000; flag=1; end;
    if YI_15900=7 then do; pvhomedY=1000001; flag=1; end;
end;

if YI_14700=-1 or YI_15900=-1 then pvhomedY=-1;
if YI_14700=-2 or YI_15900=-2 then pvhomedY=-2;
if YI_14700=-3 or YI_15900=-3 then pvhomedY=-3;
end;

/* Any mortgage or land contract on land or property */
mortgagY=0;
if YI_8400=1 or YI_8500=1 then do;
if (YI_16400=1 or YI_16400=2) and YI_16500 not in (-1, -2, -3) then mortgagY=YI_16500;
if (YI_16400=1 or YI_16400=2) and (YI_16500 eq -1 or YI_16500 eq -2 or YI_16500 eq -3) then do;
    if YI_16600=1 then do; mortgagY=12500; flag=1; end;
    if YI_16600=2 then do; mortgagY=37500; flag=1; end;
    if YI_16600=3 then do; mortgagY=75000; flag=1; end;
    if YI_16600=4 then do; mortgagY=175000; flag=1; end;
    if YI_16600=5 then do; mortgagY=375000; flag=1; end;
    if YI_16600=6 then do; mortgagY=750000; flag=1; end;
    if YI_16600=7 then do; mortgagY=1000001; flag=1; end;
end;

if YI_16400=-1 or YI_16600=-1 then mortgagY=-1;
if YI_16400=-2 or YI_16600=-2 then mortgagY=-2;
if YI_16400=-3 or YI_16600=-3 then mortgagY=-3;

```

end;

```
/* Any loans to buy or build this residence */
loanowed=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_16700=1 and YI_17000 not in (-1, -2, -3) then loanowed=YI_17000;
if YI_16700=1 and (YI_17000 eq -1 or YI_17000 eq -2 or YI_17000 eq -3) then do;
    if YI_17100=1 then do; loanowed=12500; flag=1; end;
    if YI_17100=2 then do; loanowed=37500; flag=1; end;
    if YI_17100=3 then do; loanowed=75000; flag=1; end;
    if YI_17100=4 then do; loanowed=175000; flag=1; end;
    if YI_17100=5 then do; loanowed=375000; flag=1; end;
    if YI_17100=6 then do; loanowed=750000; flag=1; end;
    if YI_17100=7 then do; loanowed=1000001; flag=1; end;
    end;
if YI_16700=-1 or YI_17100=-1 then loanowed=-1;
if YI_16700=-2 or YI_17100=-2 then loanowed=-2;
if YI_16700=-3 or YI_17100=-3 then loanowed=-3;
end;
```

```
/* Any second mortgages */
secmortY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_17200=1 and YI_17300 not in (-1, -2, -3) then secmortY=YI_17300;
if YI_17200=1 and (YI_17300 eq -1 or YI_17300 eq -2 or YI_17300 eq -3) then do;
    if YI_17400=1 then do; secmortY=2500; flag=1; end;
    if YI_17400=2 then do; secmortY=7500; flag=1; end;
    if YI_17400=3 then do; secmortY=17500; flag=1; end;
    if YI_17400=4 then do; secmortY=37500; flag=1; end;
    if YI_17400=5 then do; secmortY=75000; flag=1; end;
    if YI_17400=6 then do; secmortY=175000; flag=1; end;
    if YI_17400=7 then do; secmortY=250001; flag=1; end;
    end;
if YI_17200=-1 or YI_17400=-1 then secmortY=-1;
if YI_17200=-2 or YI_17400=-2 then secmortY=-2;
if YI_17200=-3 or YI_17400=-3 then secmortY=-3;
end;
```

```
/* Any taxes on the property to be paid */
proptaxY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_17500=1 and YI_17600 ne . and YI_17600 ge 1 then proptaxY=YI_17600;
if YI_17600=-1 then proptaxY=-1;
if YI_17600=-2 then proptaxY=-2;
if YI_17600=-3 then proptaxY=-3;
end;
```

```
/* Own a business, partnership or professional practice */
pvbussY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_17800=1 and YI_17900 not in (-1, -2, -3) then pvbussY=YI_17900;
if YI_17800=1 and (YI_17900 eq -1 or YI_17900 eq -2 or YI_17900 eq -3) then do;
    if YI_18000=1 then do; pvbussY=0; flag=1; end;
    if YI_18000=2 then do; pvbussY=12500; flag=1; end;
    if YI_18000=3 then do; pvbussY=37500; flag=1; end;
    if YI_18000=4 then do; pvbussY=75000; flag=1; end;
    if YI_18000=5 then do; pvbussY=175000; flag=1; end;
```

```

if YI_18000=6 then do; pvbussY=375000; flag=1; end;
if YI_18000=7 then do; pvbussY=750000; flag=1; end;
if YI_18000=8 then do; pvbussY=1000001; flag=1; end; end;
if YI_17800=-1 or YI_18000=-1 then pvbussY=-1;
if YI_17800=-2 or YI_18000=-2 then pvbussY=-2;
if YI_17800=-3 or YI_18000=-3 then pvbussY=-3;
end;

/* Second real estate owned */
secrestY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_18100=1 and YI_18200 not in (-1, -2, -3) then secrestY=YI_18200;
if YI_18100=1 and (YI_18200 eq -1 or YI_18200 eq -2 or YI_18200 eq -3) then do;
    if YI_18300=1 then do; secrestY=0; flag=1; end;
    if YI_18300=2 then do; secrestY=12500; flag=1; end;
    if YI_18300=3 then do; secrestY=37500; flag=1; end;
    if YI_18300=4 then do; secrestY=75000; flag=1; end;
    if YI_18300=5 then do; secrestY=175000; flag=1; end;
    if YI_18300=6 then do; secrestY=375000; flag=1; end;
    if YI_18300=7 then do; secrestY=750000; flag=1; end;
    if YI_18300=8 then do; secrestY=1000001; flag=1; end; end;
if YI_18100=-1 or YI_18300=-1 then secrestY=-1;
if YI_18100=-2 or YI_18300=-2 then secrestY=-2;
if YI_18100=-3 or YI_18300=-3 then secrestY=-3;
end;

/* Any retirement plans or pensions */
retireY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_18400=1 and YI_18500 not in (-1, -2, -3) then retireY=YI_18500;
if YI_18400=1 and (YI_18500 eq -1 or YI_18500 eq -2 or YI_18500 eq -3) then do;
    if YI_18600=1 then do; retireY=2500; flag=1; end;
    if YI_18600=2 then do; retireY=7500; flag=1; end;
    if YI_18600=3 then do; retireY=17500; flag=1; end;
    if YI_18600=4 then do; retireY=37500; flag=1; end;
    if YI_18600=5 then do; retireY=75000; flag=1; end;
    if YI_18600=6 then do; retireY=175000; flag=1; end;
    if YI_18600=7 then do; retireY=250000; flag=1; end; end;
if YI_18400=-1 or YI_18600=-1 then retireY=-1;
if YI_18400=-2 or YI_18600=-2 then retireY=-2;
if YI_18400=-3 or YI_18600=-3 then retireY=-3;
end;

/* Any savings in saving accounts, money market, funds, trusts,...*/
savingsY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_18700=1 and YI_18800 not in (-1, -2, -3) then savingsY=YI_18800;
if YI_18700=1 and (YI_18800 eq -1 or YI_18800 eq -2 or YI_18800 eq -3) then do;
    if YI_18900=1 then do; savingsY=500; flag=1; end;
    if YI_18900=2 then do; savingsY=1750; flag=1; end;
    if YI_18900=3 then do; savingsY=3750; flag=1; end;
    if YI_18900=4 then do; savingsY=7500; flag=1; end;
    if YI_18900=5 then do; savingsY=17500; flag=1; end;
    if YI_18900=6 then do; savingsY=37500; flag=1; end;

```

```

if YI_18900=7 then do; savingsY=50001; flag=1; end;
end;
if YI_18700=-1 or YI_18900=-1 then savingsY=-1;
if YI_18700=-2 or YI_18900=-2 then savingsY=-2;
if YI_18700=-3 or YI_18900=-3 then savingsY=-3;
end;

/* Any other savings in bonds or CDs */
othsavY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_19000=1 and YI_19100 not in (-1, -2, -3) then othsavY=YI_19100;
if YI_19000=1 and (YI_19100 eq -1 or YI_19100 eq -2 or YI_19100 eq -3) then do;
    if YI_19200=1 then do; othsavY=2500; flag=1; end;
    if YI_19200=2 then do; othsavY=7500; flag=1; end;
    if YI_19200=3 then do; othsavY=17500; flag=1; end;
    if YI_19200=4 then do; othsavY=37500; flag=1; end;
    if YI_19200=5 then do; othsavY=75000; flag=1; end;
    if YI_19200=6 then do; othsavY=175000; flag=1; end;
    if YI_19200=7 then do; othsavY=250001; flag=1; end;
end;
if YI_19000=-1 or YI_19200=-1 then othsavY=-1;
if YI_19000=-2 or YI_19200=-2 then othsavY=-2;
if YI_19000=-3 or YI_19200=-3 then othsavY=-3;
end;

/* Any stocks in corporations, mutual funds */
stockY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_19400=1 and YI_19500 not in (-1, -2, -3) then stockY=YI_19500;
if YI_19400=1 and (YI_19500 eq -1 or YI_19500 eq -2 or YI_19500 eq -3) then do;
    if YI_19600=1 then do; stockY=2500; flag=1; end;
    if YI_19600=2 then do; stockY=7500; flag=1; end;
    if YI_19600=3 then do; stockY=17500; flag=1; end;
    if YI_19600=4 then do; stockY=37500; flag=1; end;
    if YI_19600=5 then do; stockY=75000; flag=1; end;
    if YI_19600=6 then do; stockY=175000; flag=1; end;
    if YI_19600=7 then do; stockY=250001; flag=1; end;
end;
if YI_19400=-1 or YI_19600=-1 then stockY=-1;
if YI_19400=-2 or YI_19600=-2 then stockY=-2;
if YI_19400=-3 or YI_19600=-3 then stockY=-3;
end;

/* Present value of any vehicles owned */
pvcarsY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_19700=1 and YI_19800 not in (-1, -2, -3) then pvcarsY=YI_19800;
if YI_19700=1 and (YI_19800 eq -1 or YI_19800 eq -2 or YI_19800 eq -3) then do;
    if YI_19900=1 then do; pvcarsY=500; flag=1; end;
    if YI_19900=2 then do; pvcarsY=1750; flag=1; end;
    if YI_19900=3 then do; pvcarsY=3750; flag=1; end;
    if YI_19900=4 then do; pvcarsY=7500; flag=1; end;
    if YI_19900=5 then do; pvcarsY=17500; flag=1; end;
    if YI_19900=6 then do; pvcarsY=37500; flag=1; end;
    if YI_19900=7 then do; pvcarsY=50001; flag=1; end;
end;
if YI_19700=-1 or YI_19900=-1 then pvcars=-1;

```

```

if YI_19700=-2 or YI_19900=-2 then pvcars=-2;
if YI_19700=-3 or YI_19900=-3 then pvcars=-3;
end;

/* Amount owed on any vehicles */
owecarY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_19700=1 and YI_20000 not in (-1, -2, -3) then owecarY=YI_20000;
if YI_19700=1 and (YI_20000 eq -1 or YI_20000 eq -2 or YI_20000 eq -3) then do;
    if YI_20100=1 then do; owecarY=500; flag=1; end;
    if YI_20100=2 then do; owecarY=1750; flag=1; end;
    if YI_20100=3 then do; owecarY=3750; flag=1; end;
    if YI_20100=4 then do; owecarY=7500; flag=1; end;
    if YI_20100=5 then do; owecarY=17500; flag=1; end;
    if YI_20100=6 then do; owecarY=37500; flag=1; end;
    if YI_20100=7 then do; owecarY=50001; flag=1; end;
end;
if YI_19700=-1 or YI_20100=-1 then owecarY=-1;
if YI_19700=-2 or YI_20100=-2 then owecarY=-2;
if YI_19700=-3 or YI_20100=-3 then owecarY=-3;
end;

/* Present value of owned furniture */
PVFURNTY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_20200=1 then do; PVFURNTY=500; flag=1; end;
if YI_20200=2 then do; PVFURNTY=1750; flag=1; end;
if YI_20200=3 then do; PVFURNTY=3750; flag=1; end;
if YI_20200=4 then do; PVFURNTY=7500; flag=1; end;
if YI_20200=5 then do; PVFURNTY=17500; flag=1; end;
if YI_20200=6 then do; PVFURNTY=37500; flag=1; end;
if YI_20200=7 then do; PVFURNTY=50001; flag=1; end;
if YI_20200=-1 then PVFURNTY=-1;
if YI_20200=-2 then PVFURNTY=-2;
if YI_20200=-3 then PVFURNTY=-3;
end;

/* Any other assets not being mentioned before */
otassetY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_20300=1 and YI_20400 not in (-1, -2, -3) then otassetY=YI_20400;
if YI_20300=1 and (YI_20400 eq -1 or YI_20400 eq -2 or YI_20400 eq -3) then do;
    if YI_20500=1 then do; otassetY=2500; flag=1; end;
    if YI_20500=2 then do; otassetY=7500; flag=1; end;
    if YI_20500=3 then do; otassetY=17500; flag=1; end;
    if YI_20500=4 then do; otassetY=37500; flag=1; end;
    if YI_20500=5 then do; otassetY=75000; flag=1; end;
    if YI_20500=6 then do; otassetY=175000; flag=1; end;
    if YI_20500=7 then do; otassetY=250001; flag=1; end;
end;
if YI_20300=-1 or YI_20500=-1 then otassetY=-1;
if YI_20300=-2 or YI_20500=-2 then otassetY=-2;
if YI_20300=-3 or YI_20500=-3 then otassetY=-3;
end;

/* Any loans still owed to family or relatives */

```

```

array rloan rloan01 rloan02 rloan03 rloan04 rloan05 rloan06 rloan07 rloan08 rloan09 rloan10 rloan11 rloan12
      rloan13 rloan14 rloan15 rloan16 rloan17 rloan18 rloan19 rloan20;
array I21800 I2180001 I2180002 I2180003 I2180004 I2180005 I2180006 I2180007 I2180008 I2180009
      I2180010 I2180011 I2180012 I2180013 I2180014 I2180015 I2180016 I2180017 I2180018 I2180019
      I2180020;
array I21900 I2190001 I2190002 I2190003 I2190004 I2190005 I2190006 I2190007 I2190008 I2190009
      I2190010 I2190011 I2190012 I2190013 I2190014 I2190015 I2190016 I2190017 I2190018 I2190019
      I2190020;

do I=1 to 20;
rloan(I)=0;
if YI_8500=1 and YI_20800=1 then do;
  if I21800(I) not in (-1, -2, -3, -4) then rloan(I)=I21800(I);
  if (I21800(I) eq -1 or I21800(I) eq -2 or I21800(I) eq -3) then do;
    if I21900(I)=1 then do; rloan(I)=500; flag=1; end;
    if I21900(I)=2 then do; rloan(I)=1750; flag=1; end;
    if I21900(I)=3 then do; rloan(I)=3750; flag=1; end;
    if I21900(I)=4 then do; rloan(I)=7500; flag=1; end;
    if I21900(I)=5 then do; rloan(I)=17500; flag=1; end;
    if I21900(I)=6 then do; rloan(I)=37500; flag=1; end;
    if I21900(I)=7 then do; rloan(I)=50001; flag=1; end;
    end;
  if YI_20800=-1 or I21900(I)=-4 then rloan(I)=-1;
  if YI_20800=-2 or I21900(I)=-4 then rloan(I)=-2;
  if YI_20800=-3 or I21900(I)=-4 then rloan(I)=-3;
  if I21800(I) eq . then rloan(I)=0;
  end;
end;
/* Any other debts from loans, credit cards, etc...*/
othdebtY=0;
if YI_8400=1 or YI_8500=1 then do;
if YI_22100=1 and YI_22200 not in (-1, -2, -3) then othdebtY=YI_22200;
if YI_22100=1 and (YI_22200 eq -1 or YI_22200 eq -2 or YI_22200 eq -3) then do;
  if YI_22300=1 then do; othdebtY=500; flag=1; end;
  if YI_22300=2 then do; othdebtY=1750; flag=1; end;
  if YI_22300=3 then do; othdebtY=3750; flag=1; end;
  if YI_22300=4 then do; othdebtY=7500; flag=1; end;
  if YI_22300=5 then do; othdebtY=17500; flag=1; end;
  if YI_22300=6 then do; othdebtY=37500; flag=1; end;
  if YI_22300=7 then do; othdebtY=50001; flag=1; end;
  end;
if YI_22100=-1 or YI_22300=-1 then othdebtY=-1;
if YI_22100=-2 or YI_22300=-2 then othdebtY=-2;
if YI_22100=-3 or YI_22300=-3 then othdebtY=-3;
end;
/* Calculate the household net worth according to the youth: hhworthY=assets-liabilities */
do I=1 to 19;
hhworthY=-4;
if rloan(I) ge 0 then do;
  hhworthY=(estatesY + pvranchY + pvmobilY + pvhomeY + pbussY + secrestY + retireY + savingsY + othsavY +
  stockY + pvcarsY + PVFURNTY+otassetY) - (mortgagY + loanowed + secmortY + rloan01 + rloan02 + rloan03
  + rloan04 + rloan05 + rloan06 + rloan07 + rloan08 + rloan09 + rloan10 + rloan11 + rloan12 + rloan13 +
  rloan14 + rloan15 + rloan16 + rloan17 + rloan18 + rloan19 + othdebtY + owecarY);
  end;
end;

```

```

if YI_1900=0 or YI_8500=0 or YI_1900=-4 or YI_8500=-4 then hhworthY=-4;
if (YI_1900=1 or YI_8500=1) and (estatesY=-1 or pvranchY=-1 or pvmobilY=-1 or pvhomeY=-1 or pvbussY=-1 or
secrestY=-1 or retireY=-1 or savingsY=-1 or othsavY=-1 or stockY=-1 or pvcarsY=-1 or PVFURNTY=-1 or
otassetY=-1 or mortgagY=-1 or loanowed=-1 or secmortY=-1 or rloan01=-1 or rloan02=-1 or rloan03=-1 or
rloan04=-1 or rloan05=-1 or rloan06=-1 or rloan07=-1 or rloan08=-1 or rloan09=-1 or rloan10=-1 or rloan11=-1 or
rloan12=-1 or rloan13=-1 or rloan14=-1 or rloan15=-1 or rloan16=-1 or rloan17=-1 or rloan18=-1 or
rloan19=-1 or othdebtY=-1 or owecarY=-1) then hhworthY=-1;
if (YI_1900=1 or YI_8500=1) and (estatesY=-2 or pvranchY=-2 or pvmobilY=-2 or pvhomeY=-2 or pvbussY=-2 or
secrestY=-2 or retireY=-2 or savingsY=-2 or othsavY=-2 or stockY=-2 or pvcarsY=-2 or PVFURNTY=-2 or
otassetY=-2 or mortgagY=-2 or loanowed=-2 or secmortY=-2 or rloan01=-2 or rloan02=-2 or rloan03=-2 or
rloan04=-2 or rloan05=-2 or rloan06=-2 or rloan07=-2 or rloan08=-2 or rloan09=-2 or rloan10=-2 or rloan11=-2 or
rloan12=-2 or rloan13=-2 or rloan14=-2 or rloan15=-2 or rloan16=-2 or rloan17=-2 or rloan18=-2 or
rloan19=-2 or othdebtY=-2 or owecarY=-2) then hhworthY=-2;
if (YI_1900=1 or YI_8500=1) and (estatesY=-3 or pvranchY=-3 or pvmobilY=-3 or pvhomeY=-3 or pvbussY=-3 or
secrestY=-3 or retireY=-3 or savingsY=-3 or othsavY=-3 or stockY=-3 or pvcarsY=-3 or PVFURNTY=-3 or
otassetY=-3 or mortgagY=-3 or loanowed=-3 or secmortY=-3 or rloan01=-1 or rloan02=-3 or rloan03=-3 or
rloan04=-3 or rloan05=-3 or rloan06=-3 or rloan07=-3 or rloan08=-3 or rloan09=-3 or rloan10=-3 or rloan11=-3 or
rloan12=-3 or rloan13=-3 or rloan14=-3 or rloan15=-3 or rloan16=-3 or rloan17=-3 or rloan18=-3 or
rloan19=-3 or othdebtY=-3 or owecarY=-3) then hhworthY=-3;

/* Create gross hh income according to the youth*/
if afdcY eq -4 then afdcY=0;
if ssiy eq -4 then ssiy=0;
if othe eq -4 then othe=0;

do I=1 to 9;
groshhIY=-4;
if YI_1900=0 or YI_8500=0 or YI_1900=-4 or YI_8500=-4 then groshhIY=-4;
if (otfamI(I) ge 0 and (YI_1900=1 or YI_8500=1)) then do;
    groshhIY=(nfarmwgY + farmwgY + nfarmwgP + farmwgP + childsuY + interesY + dividend + rentalIY + pensionY
    + faincome + maincome + mgincome + fgincome + oftfamI01 + oftfamI02 + oftfamI03 + oftfamI04 + oftfamI05 +
    oftfamI06 + oftfamI07 + oftfamI08 + oftfamI09 + allowpar + allowmot + allowfat + afdcY + ssiy + othe);
end;

if (YI_1900=1 or YI_8500=1) and (nfarmwgY=-1 or farmwgY=-1 or nfarmwgP=-1 or farmwgP=-1 or childsuY=-1 or
interesY=-1 or dividend=-1 or rentalIY=-1 or pensionY=-1 or faincome=-1 or maincome=-1 or mgincome=-1 or
fgincome=-1 or oftfamI01=-1 or oftfamI02=-1 or oftfamI03=-1 or oftfamI04=-1 or oftfamI05=-1 or
otfamI06=-1 or oftfamI07=-1 or oftfamI08=-1 or oftfamI09=-1 or allowpar-1 or allowmot=-1 or allowfat=-1 or
afdcy=-1 or ssiy=-1 or othe=-1 ) then groshhIY=-3;
if (YI_1900=1 or YI_8500=1) and (nfarmwgY=-2 or farmwgY=-2 or nfarmwgP=-2 or farmwgP=-2 or childsuY=-2 or
interesY=-2 or dividend=-2 or rentalIY=-2 or pensionY=-2 or faincome=-2 or maincome=-2 or mgincome=-2 or
fgincome=-2 or oftfamI01=-2 or oftfamI02=-2 or oftfamI03=-2 or oftfamI04=-2 or oftfamI05=-2 or
otfamI06=-2 or oftfamI07=-2 or oftfamI08=-2 or oftfamI09=-2 or allowpar-2 or allowmot=-2 or allowfat=-2 or
afdcy=-2 or ssiy=-2 or othe=-2 ) then groshhIY=-3;
if (YI_1900=1 or YI_8500=1) and (nfarmwgY=-3 or farmwgY=-3 or nfarmwgP=-3 or farmwgP=-3 or childsuY=-3 or
interesY=-3 or dividend=-3 or rentalIY=-3 or pensionY=-3 or faincome=-3 or maincome=-3 or mgincome=-3 or
fgincome=-3 or oftfamI01=-3 or oftfamI02=-3 or oftfamI03=-3 or oftfamI04=-3 or oftfamI05=-3 or
otfamI06=-3 or oftfamI07=-3 or oftfamI08=-3 or oftfamI09=-3 or allowpar-3 or allowmot=-3 or allowfat=-3 or
afdcy=-3 or ssiy=-3 or othe=-3 ) then groshhIY=-3;

if YI_6000=0 then do;
if (otfamI(I) ge 0 and (YI_1900=1 or YI_8500=1)) then do;
    groshhIY=(nfarmwgY + farmwgY + nfarmwgP + farmwgP + childsuY + interesY + dividend + rentalIY + pensionY
    + faincome + maincome + mgincome + fgincome + oftfamI01 + oftfamI02 + oftfamI03 + oftfamI04 + oftfamI05 +
    oftfamI06 + oftfamI07 + oftfamI08 + oftfamI09 + allowpar + allowmot + allowfat + afdcY + ssiy + othe);
end;

```

```

if (YI_1900=1 or YI_8500=1) and (nfarmwgY=-1 or farmwgY=-1 or nfarmwgP=-1 or farmwgP=-1 or childsuY=-1 or
    interesY=-1 or dividend=-1 or rentalIY=-1 or pensionY=-1 or faincome=-1 or maincome=-1 or mgincome=-1 or
    fgincome=-1 or oftfamI01=-1 or oftfamI02=-1 or oftfamI03=-1 or oftfamI04=-1 or oftfamI05=-1 or
    oftfamI06=-1 or oftfamI07=-1 or oftfamI08=-1 or oftfamI09=-1 or allowpar=1 or allowmot=1 or allowfat=1 or
    afdcyy=1 or ssiy=1 or othe=1 ) then groshhIY=-1;
if (YI_1900=1 or YI_8500=1) and (nfarmwgY=-2 or farmwgY=-2 or nfarmwgP=-2 or farmwgP=-2 or childsuY=-2 or
    interesY=-2 or dividend=-2 or rentalIY=-2 or pensionY=-2 or faincome=-2 or maincome=-2 or mgincome=-2 or
    fgincome=-2 or oftfamI01=-2 or oftfamI02=-2 or oftfamI03=-2 or oftfamI04=-2 or oftfamI05=-2 or
    oftfamI06=-2 or oftfamI07=-2 or oftfamI08=-2 or oftfamI09=-2 or allowpar=2 or allowmot=2 or allowfat=2 or
    afdcyy=2 or ssiy=2 or othe=2 ) then groshhIY=-2;
if (YI_1900=1 or YI_8500=1) and (nfarmwgY=-3 or farmwgY=-3 or nfarmwgP=-3 or farmwgP=-3 or childsuY=-3 or
    interesY=-3 or dividend=-3 or rentalIY=-3 or pensionY=-3 or faincome=-3 or maincome=-3 or mgincome=-3 or
    income=-3 or oftfamI01=-3 or oftfamI02=-3 or oftfamI03=-3 or oftfamI04=-3 or oftfamI05=-3 or
    oftfamI06=-3 or oftfamI07=-3 or oftfamI08=-3 or oftfamI09=-3 or allowpar=3 or allowmot=3 or allowfat=3 or
    afdcyy=3 or ssiy=3 or othe=3 ) then groshhIY=-3;
end;
end;

/*correct for invalid skips*/
if (pubid=1714 or pubid=6734 or pubid=8405) then hhworthY=-3;
if (pubid=1714 or pubid=6734 or pubid=8405) then groshhIY=-3;

```

******* SECTION 3: GROSS HOUSEHOLD INCOME—PARENT AND YOUTH INFORMATION *******

/* Create the Gross household income, first looking at the parent, if not available, at the youth. Also a dummy variable indicating whether the value has been taken from the parent or youth */

```

if groshhIp ne . then do;
    hhincome=groshhIp; R=1; end;
if groshhIp eq . then do;
    hhincome=groshhIY; R=2; end;
if (groshhIp=-2 or groshhIp=-1 or groshhIp=-4 or groshhIp=-3) and groshhIY ne -4 then do;
    hhincome=groshhIY; R=2; end;

```

******* SECTION 4: POVERTY THRESHOLDS AND INCOME-TO-POVERTY THRESHOLD RATIO *******

/* The hh poverty threshold is calculated based on the total number of hh members and the number under age 18 */

```

povert=-4;
if hhmember=-4 or under18=-4 then povert=-4;
if hhmember=-1 or under18=-1 then povert=-1;
if hhmember=-2 or under18=-2 then povert=-2;
if hhmember=-3 or under18=-3 then povert=-3;
if hhmember=1 then povert=8163;

```

```

if hhmember=2 then do;
    if under18=0 then povert=10507;
    if under18=1 then povert=10815;
end;

```

```

if hhmember=3 then do;
    if under18=0 then povert=12273;
    if under18=1 then povert=12629;
    if under18=2 then povert=12641;
end;

```

```

if hhmember=4 then do;
    if under18=0 then povert=16183;
    if under18=1 then povert=16448;
    if under18=2 then povert=15911;
    if under18=3 then povert=15967;
end;

```

```

if hhmember=5 then do;
    if under18=0 then povert=19516;
    if under18=1 then povert=19800;
    if under18=2 then povert=19194;
    if under18=3 then povert=18725;
    if under18=4 then povert=18438;
end;

```

```

if hhmember=6 then do;
    if under18=0 then povert=22447;

```

```

if under18=1 then povert=22536;
if under18=2 then povert=22072;
if under18=3 then povert=21627;
if under18=4 then povert=20965;
if under18=5 then povert=20573;
end;

if hhmember=7 then do;
  if under18=0 then povert=25828;
  if under18=1 then povert=25990;
  if under18=2 then povert=25434;
  if under18=3 then povert=25046;
  if under18=4 then povert=24324;
  if under18=5 then povert=23482;
  if under18=6 then povert=22558;
end;

if hhmember=8 then do;
  if under18=0 then povert=28887;
  if under18=1 then povert=29142;
  if under18=2 then povert=28617;
  if under18=3 then povert=28158;
  if under18=4 then povert=27506;
  if under18=5 then povert=26678;
  if under18=6 then povert=25816;
  if under18=7 then povert=25597;
end;

if hhmember=9 then do;
  if under18=0 then povert=34749;
  if under18=1 then povert=34917;
  if under18=2 then povert=34453;

```

```

if under18=3 then povert=34063;
if under18=4 then povert=33423;
if under18=5 then povert=32542;
if under18=6 then povert=31746;
if under18=7 then povert=31548;
if under18=8 then povert=30333;
end;

povthr=-4;
if hhincome ge 0 then povthr=hhincome/povert;
if hhincome eq -1 or povert=-1 then povthr=-1;
if hhincome eq -2 or povert=-2 then povthr=-2;
if hhincome eq -3 or povert=-3 then povthr=-3;

povthr2=-4;
if povthr ge 0 then do;
  povthr2=povthr*100;
end;
if povthr eq -1 then povthr2=-1;
if povthr eq -2 then povthr2=-2;
if povthr eq -3 then povthr2=-3;

povthr3=-4;
if povthr2 ge 0 then do;
  povthr3=round(povthr2, 1);
end;
if povthr2 eq -1 then povthr3=-1;
if povthr2 eq -2 then povthr3=-2;
if povthr2 eq -3 then povthr3=-3;

endsas;

```

PARTICIPATION IN GOVERNMENT PROGRAMS

Variables Created: CV_AMT_GOVNT_PGM_PCY.80 – CV_AMT_GOVNT_PGM_PCY.98
 CV_GOVNT_PGM_EVER
 CV_GOVNT_PGM_YR.80 – CV_GOVNT_PGM_YR.98

Variables Used

Name in Program	Question Name on CD	Name in Program	Question Name on CD
I900_D,_M,_Y	YINF-900_D,_M,_Y	P24100	YPRG-24100
P16100	YPRG-16100	P246001M, 02M	YPRG-24600.01_M,.02_M
P16200	YPRG-16200	P246001Y, 02Y	YPRG-24600.01_Y,.02_Y
P167001M-03M	YPRG-16700.01_M-.03_M	P247001, 02	YPRG-24700.01,.02
P167001Y-03Y	YPRG-16700.01_Y-.03_Y	P248001, 02	YPRG-24800.01,.02
P170001-03	YPRG-17000.01-.03	P249001	YPRG-24900.01
P170901-03	YPRG-17090.01-.03	P249901	YPRG-24990.01
P172001M-02M	YPRG-17200.01_M-.02_M	P2500	YPRG-2500
P172001Y-02Y	YPRG-17200.01_Y-.02_Y	P252001M, 02M	YPRG-25200.01_M,.02_M
P175001-03	YPRG-17500.01-.03	P252001Y, 02Y	YPRG-25200.01_Y,.02_Y
P181001-03	YPRG-18100.01-.03	P253001, 02	YPRG-25300.01,.02
P182001	YPRG-18200.01	P254001	YPRG-25400.01
P1830011-15	YPRG-18300.01_001-_005	P259001, 02	YPRG-25900.01,.02
P1830021-25	YPRG-18300.02_001-_005	P2600	YPRG-2600
P1830031-35	YPRG-18300.03_001-_005	P260001	YPRG-26000.01
P18800	YPRG-18800	P2610011-15	YPRG-26100.01_001-_005
P18900	YPRG-18900	P2610021-25	YPRG-26100.02_001-_005
P194001M-02M	YPRG-19400.01_M-.02_M	P262001, 02	YPRG-26200.01,.02
P194001Y-02Y	YPRG-19400.01_Y-.02_Y	P264001, 02	YPRG-26400.01,.02
P195001, 02	YPRG-19500.01,.02	P26600	YPRG-26600
P196001, 02	YPRG-19600.01,.02	P26700	YPRG-26700
P197001, 02	YPRG-19700.01,.02	P2700	YPRG-2700
P197901, 02	YPRG-19790.01,.02	P272001M, Y	YPRG-27200.01_M,_Y
P200001M-02M	YPRG-20000.01_M-.02_M	P273001	YPRG-27300.01
P200001Y-02Y	YPRG-20000.01_Y-.02_Y	P274001	YPRG-27400.01
P201001, 02	YPRG-20100.01,.02	P275001	YPRG-27500.01
P202001	YPRG-20200.01	P275901	YPRG-27590.01
P207001, 02	YPRG-20700.01,.02	P278001M, Y	YPRG-27800.01_M,_Y
P208001	YPRG-20800.01	P279001	YPRG-27900.01
P2090011-15	YPRG-20900.01_001-_005	P2800	YPRG-2800
P2090021-25	YPRG-20900.02_001-_005	P285001	YPRG-28500.01
P210001, 02	YPRG-21000.01,.02	P286001	YPRG-28600.01
P212001, 02	YPRG-21200.01,.02	P287001	YPRG-28700.01
P21400	YPRG-21400	P289001	YPRG-28900.01
P21500	YPRG-21500	P2900	YPRG-2900
P220001M-03M	YPRG-22000.01_M-.03_M	P3000	YPRG-3000
P220001Y-03Y	YPRG-22000.01_Y-.03_Y	P30500	YPRG-30500
P221001-03	YPRG-22100.01-.03	P3100	YPRG-3100
P222001-03	YPRG-22200.01-.03	P310001M, Y	YPRG-31000.01_M,_Y
P223001-03	YPRG-22300.01-.03	P311001	YPRG-31100.01
P223901-03	YPRG-22390.01-.03	P312001	YPRG-31200.01
P226001M-02M	YPRG-22600.01_M-.02_M	P313001	YPRG-31300.01
P226001Y-02Y	YPRG-22600.01_Y-.02_Y	P313901	YPRG-31390.01
P227001, 02	YPRG-22700.01,.02	P316001M, Y	YPRG-31600.01_M,_Y
P228001	YPRG-22800.01	P317001	YPRG-31700.01
P230001-03	YPRG-23000.01-.03	P318001	YPRG-31800.01
P233001-03	YPRG-23300.01-.03	P3200	YPRG-3200
P234001	YPRG-23400.01	P323001	YPRG-32300.01
P2350011-15	YPRG-23500.01_001-_005	P324001	YPRG-32400.01

Appendix 5: Income and Assets Variable Creation

P2350021-25	YPRG-23500.02_001-_005	P3250011-15	YPRG-32500.01_001-_005
P2350031-35	YPRG-23500.03_001-_005	P326001	YPRG-32600.01
P236001-03	YPRG-23600.01-.03	P329001	YPRG-32900.01
P238001, 02	YPRG-23800.01, .02	P3300	YPRG-3300
P2400	YPRG-2400	PUBID	PUBID
P24000	YPRG-24000	sKEY_D, _M, _Y	KEY!BDATE_D, _M, _Y

This program creates several variables describing the respondent's participation in government programs for the economically disadvantaged. During the interview, respondents report amounts received and months of participation in Aid to Families with Dependent Children (AFDC); food stamps; the Low Income Home Energy Assistance Program (LIHEAP); Supplemental Security Income (SSI); and Women, Infants, and Children (WIC). There is also an "other assistance" question to capture information about any other government program from which respondents may have received assistance. Users should note that information about unemployment compensation and worker's compensation is collected in separate question series. Participation in these programs is **not** included here but will be summarized in separate variables in future rounds (in round 1, no respondents reported participation in these programs).

***** The program to create these variables first creates a month-by-month participation array for each of the six categories (AFDC, food stamps, LIHEAP, SSI, WIC, and other). These month-by-month variables constitute part of the event history array for program participation; see appendix 7 for more information. Due to space considerations, only the AFDC array creation is shown here in its entirety; the arrays for the other five programs are created in a nearly identical manner. Notes in this program identify differences in hand edits among the six arrays; otherwise, all information is the same. Users who need access to the complete code should contact NLS User Services. After all six arrays are created, the program merges data from the six categories to create the summary variables. *****

***** SECTION 1: AFDC MONTH-BY-MONTH PARTICIPATION ARRAY *****

```
%let num=3; /* maximum # of spells that R reports participating in AFDC */
%let years=19; /* # of years covered by array */
%let name=afdc;
```

***** This portion of the program changes any amounts that exceed the yearly AFDC limit (federal, not program specific). In addition, a flag (amtvar) is created to let users know if the amount is imputed or not. The values are the following: 0=no change to the reported value
 1=reported value too high, imputed value used
 2=reported value too high, reported value used *****;

```
amtvar1=-4; amtvar2=-4; amtvar3=-4;
```

```
if pubid=5296 then do;
  p181001=ceil(3126/12);
  amtvar1=1;
end;
```

***** The hand edits for the other programs are as follows:

```
Food stamps      if pubid=7911 then do; P207001=ceil(12000/28); amtvar1=1; end;
                  if pubid=5296 then do; P207001=ceil(2616/21); amtvar1=1; end;
LIHEAP, SSI, WIC, and other have no hand edits. *****
```

***** The variables created and used in this program are put into an array using the command below. Each array is defined by the (&num) argument where &num refers to the number of loops in the array. *****;

```
array syear(&num) ybeg1-ybeg&num;
array eyear(&num) yend1-yend&num;
array smonth(&num) mbeg1-mbeg&num;
array emonth(&num) mend1-mend&num;
```

```

array sdate(&num) sdate1-sdate&num;
array edate(&num) edate1-edate&num;
array flag(&num) flag1-flag&num;
array censor(&num) censor1-censor&num;
array &name(&years) &name.1-&name.&years;

array weeks(&num) p175001-p17500&num;           /*answer number of weeks*/
array rev(&num) p170001-p17000&num;              /*still receiving*/

array money(&num) p181001-p18100&num;           /*average amount per month*/
array estmoney(&num) p182001-p18200&num;          /*estimated ave amount per month*/
array amount(&num) amount1-amount&num;

array oldsmon(&num) p167001m p167002m p167003m;    /*original start month*/
array oldemon(&num) p172001m p172002m p172003m;    /*original end month*/

array oldsyear(&num) p167001y p167002y p167003y;   /*original start year*/
array oldeyear(&num) p172001y p172002y p172003y;   /*original end year*/

array flagvar(&num) flagvar1 flagvar2 flagvar3;      /*variable indicated imputed participation values*/
array amtvar(&num) amtvar1 amtvar2 amtvar3;          /* variable indicating imputed amounts*/

array cstart(&num) cstart1 cstart2 cstart3;
array cstop(&num) cstop1 cstop2 cstop3;

***** These two-dimensional arrays create variables for each year and month listed (starting from the birth month of the oldest NLSY97 respondent and ending with the most recent interview month). *****;

array count(&years,12)
  &pre._80_1-&pre._80_12  &pre._81_1-&pre._81_12  &pre._82_1-&pre._82_12  &pre._83_1-&pre._83_12
  &pre._84_1-&pre._84_12  &pre._85_1-&pre._85_12  &pre._86_1-&pre._86_12  &pre._87_1-&pre._87_12
  &pre._88_1-&pre._88_12  &pre._89_1-&pre._89_12  &pre._90_1-&pre._90_12  &pre._91_1-&pre._91_12
  &pre._92_1-&pre._92_12  &pre._93_1-&pre._93_12  &pre._94_1-&pre._94_12  &pre._95_1-&pre._95_12
  &pre._96_1-&pre._96_12  &pre._97_1-&pre._97_12  &pre._98_1-&pre._98_12;

array countamt(&years,12)
  a&pre._80_1-a&pre._80_12  a&pre._81_1-a&pre._81_12  a&pre._82_1-a&pre._82_12
  a&pre._83_1-a&pre._83_12  a&pre._84_1-a&pre._84_12  a&pre._85_1-a&pre._85_12
  a&pre._86_1-a&pre._86_12  a&pre._87_1-a&pre._87_12  a&pre._88_1-a&pre._88_12
  a&pre._89_1-a&pre._89_12  a&pre._90_1-a&pre._90_12  a&pre._91_1-a&pre._91_12
  a&pre._92_1-a&pre._92_12  a&pre._93_1-a&pre._93_12  a&pre._94_1-a&pre._94_12
  a&pre._95_1-a&pre._95_12  a&pre._96_1-a&pre._96_12  a&pre._97_1-a&pre._97_12
  a&pre._98_1-a&pre._98_12;

array a_&name(&years) a_&name.1-a_&name.&years;
array p_&name(&years) p_&name.1-p_&name.&years;

***** The first variable 'receive' below indicates whether the person reports participation in the program at the time of the interview. If the second variable 'noeligh' equals 1, it indicates that the respondent is not eligible to receive benefits from the program. *****;

receive=p16100;

if p2400=1 then rec=1; else rec=0;
if p2400=-4 then noeligh=1; else noeligh=0;

```

Appendix 5: Income and Assets Variable Creation

***** Use the category reported by the respondent to create an estimated amount. The estimated amount is the midpoint rounded down. Note that the 12th category lists \$1251 as the amount. This amount was chosen since the category is unbounded—the number represents one dollar above the lower bound. *****;

```
do i = 1 to &num;
if money(i) in (-1, -2, -3) then do;
  if estmoney(i)= 1 then amount(i)=50;           else if estmoney(i)=2 then amount(i)=150;
  else if estmoney(i)=3 then amount(i)=250;       else if estmoney(i)=4 then amount(i)=350;
  else if estmoney(i)=5 then amount(i)=450;       else if estmoney(i)=6 then amount(i)=550;
  else if estmoney(i)=7 then amount(i)=650;       else if estmoney(i)=8 then amount(i)=750;
  else if estmoney(i)=9 then amount(i)=850;       else if estmoney(i)=10 then amount(i)=950;
  else if estmoney(i)=11 then amount(i)=1125;     else if estmoney(i)=12 then amount(i)=1251;
  else if estmoney(i) in (-1,-2,-3) then amount(i)=estmoney(i);
end;
else amount(i)=money(i);
end;
```

***** This portion of the SAS program defines the start date and end dates. If the respondent reports still receiving, the interview date is used as the temporary end date for the last loop reported. In the next survey round, the respondent will be asked if he or she is still receiving and, if not, a permanent end date equivalent to the interview date of the previous round will be assigned. Users will be able to tell which method was used by looking at the following participation flag variable created during the program. The categories are the following:

```
1=respondent reported participation dates
2=start month imputed
3=start month and year imputed
4=stop month imputed
5=stop month and year imputed
6=start and stop dates imputed *****;
```

```
do i=1 to &num;
eyear(i)=-4;                      /*initialize values*/
emonth(i)=-4;                     /*initialize values*/
flagvar(i)=-4;                    /*initialize values*/
smonth(i)=oldsmon(i);             /*this creates a new array smonth so that oldsmon is not overwritten*/
syear(i)=oldsyear(i);              /*this creates a new array syear so that oldsyear is not overwritten*/
if rev(i)=0 then do;               /*if no interruption reported and still receiving (rev(i)=0) then use int. date*/
  emonth(i)=i900_m;                /*assigns temporary end date to the last loop*/
  eyear(i)=i900_y;                 /*assigns temporary end date to the last loop*/
  flagvar(i)=1;
end;
else do;
  emonth(i)=oldemon(i);           /*if an interruption is reported, use the reported date*/
  eyear(i)=oldeyear(i);
  flagvar(i)=1;
if i=&num and oldeyear(i)=. then do; /*an end variable is created when, for example, there are 3
                                         start loops reported in the data but only 2 end loops reported
                                         (the respondents in the third loop are all still receiving)*/
  emonth(i)=-4;
  eyear(i)=-4;
  flagvar(i)=-4;
end;
end;
end;

do i=1 to &num;
if eyear(i)=-4 then do;
  flagvar(i)=-4;
end;
end;
```

```

*****Count the number of loops for each respondent. This is used later in the SAS program.*****
loopnum=&num;
do i = &num to 1 by -1; /*Starts from the maximum loops and subtracts 1 for each loop with a valid skip*/
if smonth(i) = -4 and syear(i)=-4 then loopnum=i-1;
end;

/**Create variable indicating censoring; create flag indicating the imputation in dates. NOTE: Not in the keep
statement**/

do i=1 to &num;
flag(i)=-4;
censor(i)=-4;
if loopnum >0 then censor(i)=0;
if rev(i)=0 then censor(i)=1;
end;

do i=1 to &num ;
weekmon =ceil(weeks(i)/4.3);                                /**Gives a month count by ceiling the weeks/4.3***/

*****1. if start year is known and month is unknown ****/
if (-3 <= smonth(i) <= -1) and (syear(i) > 0 ) and (weeks(i)>0) then do; /*mon. missing, yr. present, wks. known*/
    flag(i)=smonth(i);                                         /*This flag will be a -1 or -2 if month is missing*/
    flagvar(i)=2;

/** 1.a' if weeks are known, then count forward by the number of weeks from January of the start year if not
currently receiving. If the count exceeds the interview date then stop counting at the interview date. ****/
if receive=0 then do;
    smonth(i)=1;
    eyear(i)= int((syear(i)*12 + smonth(i) + weekmon -1)/12);
    emonth(i) = mod((syear(i)*12 + smonth(i) + weekmon -1),12);
    if emonth(i)=0 then do;
        emonth(i)=12;
        eyear(i)=eyear(i)-1;
    end;
    if (eyear(i)*12 + emonth(i)) > (i900_y*12 +i900_m) then do;           /*If the year and month extracted above
exceeds the interview date then replace
with the interview date. */
        emonth(i)=i900_m;
        eyear(i)=i900_y;
    end;
end;

/** 1.a'' If weeks are known and R currently receiving, then count backwards by the number of weeks from the
interview date. If the number of weeks falls short of the start year, the start month is December of that year.
If the number of weeks is past the start year, then the start month is January of that year.****/
else if receive=1 then do;
    emonth(i)=i900_m;                                         /*This sets an end date equal to the interview date */
    eyear(i)=i900_y;
    temp_y=int((i900_y*12+i900_m - weekmon + 1)/12);
    smonth(i)=mod((i900_y*12 + i900_m - weekmon + 1 ),12);
    if smonth(i)=0 then do;
        smonth(i)=12;
        temp_y=temp_y-1;
    end;
    if syear(i) > temp_y then smonth(i) = 1;                  /*If the calculated year is after the temp year, then end the
array in January of the calculated year.*/
    if syear(i) < temp_y then smonth(i) = 12;                /*If the calculated year is before the temp year,
then end the array in December of the calculated year.*/

```

```

    end;
    end;

    /*** 1.b Start year is known, weeks and start month are unknown *****/
    if (-3 <= smonth(i) <-1) and (syear(i) >0 ) and (weeks(i)<0) then do;
        flag(i)=smonth(i);
        flagvar(i)=2;

    /**Start month is January**/
    smonth(i)=1;
    if receive=1 then do;           /*if currently receiving then replace end month and year with interview date*/
        emonth(i)=i900_m;
        eyear(i)=i900_y;
    end;
    else if receive=0 then do;     /*if not currently receiving then use December of the start year*/
        emonth(i)=12;
        eyear(i)=syear(i);

    /** Cut off the array at the interview date if start year is interview year and December is later than interview date */
    if eyear(i)=i900_y and emonth(i) > i900_m then emonth(i)=i900_m;
    end;
end;

/****2. if start year is unknown but weeks are known then count back from interview date if currently receiving. If
not currently receiving, then count back from interview date to find the most recent year the respondent could have
begun receiving and receive for that number of months; then count forward the number of months from January of
that year ****/
if weeks(i)>0 and syear(i)<0 then do;
    flag(i)=syear(i);
    flagvar(i)=3;
    if receive=1 then do;
        emonth(i)=i900_m;
        eyear(i)=i900_y;
        smonth(i)=mod((eyear(i)*12 + emonth(i) - weekmon +1),12);
        syear(i)=int((eyear(i)*12 + emonth(i) - weekmon +1)/12);
        if smonth(i)=0 then do;
            smonth(i)=12;
            syear(i)=syear(i)-1;
        end;
    end;
    if receive=0 then do;
        smonth(i)=1;
        syear(i)=int((i900_y*12 + i900_m - weekmon+1)/12);
        if mod((i900_y*12 + i900_m - weekmon+1),12) = 0 then syear(i)=syear(i)-1;
        eyear(i)=int((syear(i)*12 + smonth(i)+weekmon-1)/12);
        emonth(i) = mod((syear(i)*12 + smonth(i)+weekmon-1),12);
        if emonth(i)=0 then do;
            emonth(i)=12;
            eyear(i)=eyear(i)-1;
        end;
    end;
end;

/****3. If stop year is known and weeks are known and not currently receiving, but stop month is not known, then
count forward from start year. If the number of months falls short of the stop year, then use January of the end year

```

as the stop date; if the number of months exceeds the stop year, then end the array in the December of the stop year. If the stop year is equal to the interview year and the stop month exceeds the interview month, then stop at the interview date. If currently receiving, use interview date as the stop date. *****/

```
if (-3 <= emonth(i) < -1) and (eyear(i) > 0) and (weeks(i) > 0) then do;
```

```
    flag(i)=emonth(i);
```

```
    flagvar(i)=4;
```

```
    if receive=1 then do;
```

```
        emonth(i)=i900_m;
```

```
        eyear(i)=i900_y;
```

```
        end;
```

```
    if receive=0 then do;
```

```
        temp_y=int((syear(i)*12 + smonth(i) + weekmon-1)/12);
```

```
        emonth(i)=mod((syear(i)*12+smonth(i)+weekmon -1),12);
```

```
        if emonth(i)=0 then do;
```

```
            emonth(i)=12;
```

```
            temp_y=temp_y-1;
```

```
            end;
```

```
            if temp_y < eyear(i) then emonth(i)=1;
```

```
            if temp_y > eyear(i) then emonth(i)=12;
```

```
            if (eyear(i) = i900_y )and (emonth(i) > i900_m) then do;
```

```
                emonth(i)=i900_m;
```

```
                end;
```

```
            end;
```

```
        end;
```

****4. If stop year is known, but weeks are unknown and stop month is unknown *****/

```
if (-3 <= emonth(i) < -1) and (eyear(i) > 0) and (weeks(i) < 0) then do;
```

```
    flag(i)= emonth(i);
```

```
    flagvar(i)=4;
```

```
    if receive=1 then do;
```

```
        emonth(i)=i900_m;
```

```
        eyear(i)=i900_y;
```

```
        end;
```

```
    if receive=0 then emonth(i)=12;
```

```
    if receive=0 and eyear(i)=i900_y then emonth(i)=i900_m;
```

```
end;
```

*** 5. if stop year is unknown, and weeks are unknown***/

```
if (-3 <= eyear(i)<=-1) and (weeks(i) < 0 ) then do;
```

```
    flag(i)=eyear(i);
```

```
    flagvar(i)=5;
```

```
    if receive=1 then do;
```

```
        emonth(i)=i900_m;
```

```
        eyear(i)=i900_y;
```

```
        end;
```

```
    if receive=0 then do;
```

```
        eyear(i)=syear(i);
```

```
        emonth(i)=12;
```

```
        if syear(i)=i900_y then emonth(i)=i900_m;
```

```
        end;
```

```
    end;
```

****6. if stop year is unknown, but weeks are known****/

```
if (-3 <= eyear(i)<=-1) and (weeks(i) > 0 ) then do;
```

```
    flag(i)=eyear(i);
```

```
    flagvar(i)=5;
```

```
    if receive=1 then do;
```

Appendix 5: Income and Assets Variable Creation

```
emonth(i)=i900_m;
eyear(i)=i900_y;
end;
if receive=0 then do;
    eyear(i)=int((syear(i)*12 + smonth(i)+weekmon-1)/12);
    emonth(i)=mod((syear(i)*12 + smonth(i) + weekmon -1),12);
    if emonth(i)=0 then do;
        emonth(i)=12;
        eyear(i)=eyear(i)-1;
        end;
    if (eyear(i)*12 + emonth(i)) > (i900_y*12 +i900_m) then do;
        eyear(i)=i900_y;
        emonth(i)=i900_m;
        end;
    end;
end;
****7. if start year, stop year, and weeks are unknown, use the date since last interview to interview date (in
R1=birthdate to interview date)****/
if (-3 =< syear(i) <= -1) and (-3 =< weeks(i) <= -1 ) and (-4 =< eyear(i) <= -1) then do;
flag(i)=syear(i);
flagvar(i)=6;
emonth(i)=i900_m;
eyear(i)=i900_y;
smonth(i)=skey_m;
syear(i)=skey_y;
end;

****8. if start year and month unknown, weeks never asked and stop date (year and month) known****/
if (-3 =< syear(i) <= -1) and (-4 =< weeks(i) <= -1 ) and (eyear(i) >0) then do;
flag(i)=syear(i);
flagvar(i)=3;
eyear(i)=oldeyear(i);
emonth(i)=oldemon(i);
smonth(i)=skey_m;
syear(i)=skey_y;
end;
end;

****9. if invalidly skipped from section - ROUND 1 ONLY ****/
do i=1 to &num;
if pubid=1714 or pubid=6734 or pubid=8405 then do;
flag(i)=-3;
flagvar(i)=-3;
amtvar(i)=-3;
emonth(i)=i900_m;
eyear(i)=i900_y;
smonth(i)=skey_m;
syear(i)=skey_y;
end;
end;

*****INITIALIZE EVENT HISTORY VARIABLES*****
do M=1 to &years;
do N = 1 to 12;
if noelgb=1 then count(m,n)=-4;
else count(M,N)=0;
```

```

*   if (m*12+n) > ((i900_y-1979)*12 +i900_m) then count(m,n)=-4;
  countamt(m,n)=-4;
end;
end;

*****LOOP 1 STARTS*****/

if loopnum^=0 then do;

***** to get the start and end dates with the format as ddmmmyyy*****
Do i = 1 to loopnum;
  do j=1 to 2;
    sdate(i)=INPUT('01'||TRIM(LEFT(put(smonth(i),z2.0)))||trim(left(syear(i))),ddmmyy8.);
    edate(i)=INPUT('01'||TRIM(LEFT(put(emonth(i),z2.0)))||trim(left(eyear(i))),ddmmyy8.);
  end;
end;

format sdate1 edate1 sdate2 edate2 sdate3 edate3 ddmmyy8.;

*****Assign values to event history and amount variables*****
do i= 1 to loopnum;
  do K=sdate(i) to edate(i);
    mmm=month(K);
    yyy=year(K)-1979;
    if flag(i) in (-1,-2) then do;      /**the flags signal that the date has been imputed earlier in the program**/
      count(yyy,mmm)=flag(i);
      countamt(yyy,mmm)=amount(i);
    end;
    else do;
      count(yyy,mmm)=1;           /**lays down a '1' in the array for each month received**/
      countamt(yyy,mmm)=amount(i);   /**puts in the corresponding amount for each month received**/
    end;
    end;
  end;
end;

do i=1 to &num;
  if flag(i)=-3 then do;
    do j=1 to 2;
      sdate(i)=INPUT('01'||TRIM(LEFT(put(smonth(i),z2.0)))||trim(left(syear(i))),ddmmyy8.);
      edate(i)=INPUT('01'||TRIM(LEFT(put(emonth(i),z2.0)))||trim(left(eyear(i))),ddmmyy8.);
    end;
  end;
end;

format sdate1 edate1 sdate2 edate2 sdate3 edate3 ddmmyy8.;

do K=sdate(i) to edate(i);
  mmm=month(i);
  yyy=year(k)-1979;
  count(yyy,mmm)=flag(i);
  countamt(yyy,mmm)=flag(i);
end;
end;
end;

*****assign continuous month value to start and stop dates*****
cstart1=-4; cstart2=-4; cstart3=-4; cstop1=-4; cstop2=-4; cstop3=-4;

```

```

do i=1 to loopnum;
  cstart(i)=((12*(syear(i)-1980))+smonth(i));
  cstop(i)=((12*(eyear(i)-1980))+emonth(i));
end;

do i=1 to &num;
  if flagvar(i)=-3 then do;
    cstart(i)=-3;
    cstop(i)=-3;
  end;
end;

*****END OF LOOP 1*****/

/**initialize to '0' the number of months and the amounts received, by years and the total over all years ***/
do i=1 to &years;
  &name(i)=0; a_&name(i)=0;
end;
&name.all=0; a&pre._all=0;

m1=0;
m2=0;
**** calculate the amount of months in total as well in each year receiving benefits ****/
do i=1 to &years;
  n1=0;
  n2=0;
  do j= 1 to 12;
    if count(i,j) >=0 then do;
      &name(i)=&name(i) + count(i,j);
      &name.all=&name.all + count(i,j);
      n1=n1+1;                                /**the n1 refers to the number of months receiving in a year**/
      m1=m1+1;                                /**the total number of month reveiving in a year****/
    end;
    if countamt(i,j)>=0 then do;           /**NOTE: Rs who do not report receiving are assigned a '-4'**/ 
      a_&name(i)=a_&name(i) + countamt(i,j);
      a&pre._all = a&pre._all +countamt(i,j);
      n2=n2+1;                                /**the n2 refers to the # of months received money in a year**/
      m2=m2+1;                                /**the total # of months received money over all years***/
    end;
    if n1 = 0 then &name(i) = -4;
    if n2 = 0 then a_&name(i) = -4;           /**if didn't participate in a program then set equal to '-4'**/ 
  end;
  if m1=0 then &pre.all=-4;
  if m2 = 0 then a&pre._all=-4;            /**if didn't participate in any years then total set equal to '-4'***/

do i=1 to &years;
  do j=1 to 12;
    if count(i,j) in (-1,-2,-3) then do;   /**when the data have been imputed then give the '-1' or '-2' value to
      &name(i)=count(i,j);                  the total for each year and overall**/
      &name.all=count(i,j);
    end;
    if countamt(i,j) in (-1,-2,-3) then do;
      a_&name(i)=countamt(i,j);
      a&pre._all=countamt(i,j);
    end;
  end;

```

```

end;

*****age14 determination - subtract 1 month so that first month eligible is month turned 14*****
if skey_m>1 then do;
    age14_m=(skey_m-1); age14_d=skey_d; age14_y=skey_y+14;
end;
if skey_m=1 then do;
    age14_m=12; age14_d=skey_d; age14_y=(skey_y+13);
end;

age14dt=INPUT('01'||TRIM(LEFT(put(age14_m,z2.0)))||trim(left(age14_y)),ddmmyy8.);
format age14dt ddmmyy8.;

*****initial date*****
init_m=1; init_d=1; init_y=1980;
init=INPUT('01'||TRIM(LEFT(put(init_m,z2.0)))||trim(left(init_y)),ddmmyy8.);
format init ddmmyy8.;

*****interview date*****
if i900_m<12 then do;
    inter_m=i900_m+1; inter_d=i900_d; inter_y=i900_y;
end;
if i900_m=12 then do;
    inter_m=i900_m; inter_d=i900_d; inter_y=i900_y+1;
end;
inter=INPUT('01'||TRIM(LEFT(put(inter_m,z2.0)))||trim(left(inter_y)),ddmmyy8.);
format inter ddmmyy8.;

*****end date - latest possible month*****
end_m=12; end_d=31; end_y=1998;

enddate=INPUT('01'||TRIM(LEFT(put(end_m,z2.0)))||trim(left(end_y)),ddmmyy8.);
format enddate ddmmyy8.;

*****truncate arrays to the month of the youth's 14th birthday and forward*****
do i=init to age14dt;
    MMM=MONTH(i); YYY=YEAR(i)-1979;
    if age14dt>init then do;
        count(yyy,mmm)=-4; countamt(yyy,mmm)=-4;
    end;
end;

*****turn arrays past interview date to -5s*****
do i=inter to enddate;
    MMM=MONTH(i); YYY=YEAR(i)-1979;
    if enddate>inter then do;
        count(yyy,mmm)=-5; countamt(yyy,mmm)=-5;
    end;
end;

*****turns those not eligible in round 1 to -4s*****
if AFDCALL=0 and noelg=1 then AFDCALL=-4;

****this is to get past the problems found in unclean data***
do i=1 to &years;
    if rec=0 then &name(i)=0;
    if rec=0 then a_&name(i)=-4;

```

```

end;
if rec=0 then AFDCALL=0;
if rec=0 then AA_ALL=-4;

*****correct for R1 invalid skips*****
do i=1 to 19;
  if (pubid=1714 or pubid=6734 or pubid=8405) then &name(i)=-3;
  if (pubid=1714 or pubid=6734 or pubid=8405) then a_&name(i)=-3;
  if (pubid=1714 or pubid=6734 or pubid=8405) then AFDCALL=-3;
  if (pubid=1714 or pubid=6734 or pubid=8405) then AA_ALL=-3;
end;

***** Hand edits for invalid skips in other five arrays:
Food stamps do i=1 to 19;
  if pubid in (1714, 6734, 8405) then &name(i)=-3 and a_&name(i)=-3 and FOODCALL=-3 and
    AF_ALL=-3; end;
LIHEAP do i=to 19;
  if pubid in (1714, 6734, 8405) then &name(i)=-3 and a_&name(i)=-3 and LIHALL=-3 and
    AL_ALL=-3; end;
SSI do i=to 19;
  if pubid in (1714, 6734, 8405) then &name(i)=-3 and a_&name(i)=-3 and SSIALL=-3 and
    AS_ALL=-3; end;
WIC do i=to 19;
  if pubid in (1714, 6734, 8405) then &name(i)=-3 and a_&name(i)=-3 and WICALL=-3 and
    AW_ALL=-3; end;
Other do i=to 19;
  if pubid in (1714, 6734, 8405) then &name(i)=-3 and a_&name(i)=-3 and OTHALL=-3 and
    AO_ALL=-3; end; *****

```

***** This ends the creation of the AFDC month-by-month array *****

***** SECTION 2: SUMMARY VARIABLE CREATION—AMOUNT OF MONEY RECEIVED *****

```
%let pre1=a; %let pre2=f; %let pre3=l; %let pre4=s; %let pre5=o; %let pre6=w;
```

```
merge afdc food ssi wic lih other;
by pubid;
```

```
array a&pre1(19,12)
  a&pre1._80_1-a&pre1._80_12  a&pre1._81_1-a&pre1._81_12  a&pre1._82_1-a&pre1._82_12
  a&pre1._83_1-a&pre1._83_12  a&pre1._84_1-a&pre1._84_12  a&pre1._85_1-a&pre1._85_12
  a&pre1._86_1-a&pre1._86_12  a&pre1._87_1-a&pre1._87_12  a&pre1._88_1-a&pre1._88_12
  a&pre1._89_1-a&pre1._89_12  a&pre1._90_1-a&pre1._90_12  a&pre1._91_1-a&pre1._91_12
  a&pre1._92_1-a&pre1._92_12  a&pre1._93_1-a&pre1._93_12  a&pre1._94_1-a&pre1._94_12
  a&pre1._95_1-a&pre1._95_12  a&pre1._96_1-a&pre1._96_12  a&pre1._97_1-a&pre1._97_12
  a&pre1._98_1-a&pre1._98_12 ;
```

**** a&pre2, a&pre3, a&pre4, a&pre5, and a&pre6, are then arrayed in identical fashion ****

```
array account(19,12)
  at_80_1-at_80_12  at_81_1-at_81_12  at_82_1-at_82_12  at_83_1-at_83_12  at_84_1-at_84_12
  at_85_1-at_85_12  at_86_1-at_86_12  at_87_1-at_87_12  at_88_1-at_88_12  at_89_1-at_89_12
  at_90_1-at_90_12  at_91_1-at_91_12  at_92_1-at_92_12  at_93_1-at_93_12  at_94_1-at_94_12
  at_95_1-at_95_12  at_96_1-at_96_12  at_97_1-at_97_12  at_98_1-at_98_12 ;
```

```
array atotal(19) atotal80 atotal81 atotal82 atotal83 atotal84 atotal85 atotal86 atotal87 atotal88 atotal89 atotal90
atotal91 atotal92 atotal93 atotal94 atotal95 atotal96 atotal97 atotal98;
```

```

do i=1 to 19;
  do j=1 to 12;
    acount(i,j)=0;
    n=0;
    do k=a&pre1(i,j), a&pre2(i,j), a&pre3(i,j), a&pre4(i,j), a&pre5(i,j), a&pre6(i,j);
      if k >=0 then do;
        acount(i,j)= acount(i,j) + k;
        n=n+1;
      end;
    end;
    if n=0 then acount(i,j)=-4;
    do k=a&pre1(i,j), a&pre2(i,j), a&pre3(i,j), a&pre4(i,j), a&pre5(i,j), a&pre6(i,j);
      if k =-2 and acount(i,j) ^= -1 then acount(i,j)= k;
      if k=-1 then acount(i,j)= k;
    end;
  end;
end;

/**initialize amount of months received, by years and total *****/
do i=1 to 19;
  atotal(i)=0;
end;
atall=0;
m=0;

**** calculate the amount of months in total as well in each year receiving AFDC****/
do i=1 to 19;
  n=0;
  do j= 1 to 12;
    if acount(i,j) >=0 then do;
      atotal(i)=atotal(i) + count(i,j);
      atall=atall + count(i,j);
      n=n+1;
      m=m+1;
    end;
  end;
  if n = 0 then atotal(i)=-4;
end;
if m=0 then atall=-4;

do i=1 to 19;
  do j=1 to 12;
    if acount(i,j) in (-1,-2,-3) and atotal(i) ^= -1 then do;
      atotal(i)=acount(i,j);
      atall=acount(i,j);
    end;
  end;
end;

/*correct for invalid skips*/
do i=1 to 19;
  if (pubid=1714 or pubid=6734 or pubid=8405) then atotal(i)=-3 and atall=-3;;
end;

***** SECTION 3: SUMMARY VARIABLE CREATION—MONTHS RECEIVED PAYMENT *****

/**total event history array***/
```

```

array count(19,12)
  t_80_1-t_80_12  t_81_1-t_81_12  t_82_1-t_82_12  t_83_1-t_83_12  t_84_1-t_84_12
  t_85_1-t_85_12  t_86_1-t_86_12  t_87_1-t_87_12  t_88_1-t_88_12  t_89_1-t_89_12
  t_90_1-t_90_12  t_91_1-t_91_12  t_92_1-t_92_12  t_93_1-t_93_12  t_94_1-t_94_12
  t_95_1-t_95_12  t_96_1-t_96_12  t_97_1-t_97_12  t_98_1-t_98_12 ;

array total(19) total80 total81 total82 total83 total84 total85 total86 total87 total88 total89 total90 total91
total92 total93 total94 total95 total96 total97 total98;

/**lists the codes using a priority ranking for programs - for example, (-4, 0, -2, -1, 1) 1 overwrites everything,
-4 is always overwritten, a 0 is overwritten by an unknown answer of -1 or -2. A -3.5 replaces 0 temporarily so that
the priority ranking is easily done in order - then it is changed back to 0**/ 

do i=1 to 19;
  do j=1 to 12;
    if &pre1(i,j)=0 then &pre1(i,j)=-3.5;
    if &pre2(i,j)=0 then &pre2(i,j)=-3.5;
    if &pre3(i,j)=0 then &pre3(i,j)=-3.5;
    if &pre4(i,j)=0 then &pre4(i,j)=-3.5;
    if &pre5(i,j)=0 then &pre5(i,j)=-3.5;
    if &pre6(i,j)=0 then &pre6(i,j)=-3.5;
  count(i,j)= max(&pre1(i,j), &pre2(i,j), &pre3(i,j), &pre4(i,j), &pre5(i,j), &pre6(i,j));
  if count(i,j)=-3.5 then count(i,j)=0;
  if &pre1(i,j)=-3.5 then &pre1(i,j)=0 ;
  if &pre2(i,j)=-3.5 then &pre2(i,j)=0;
  if &pre3(i,j)=-3.5 then &pre3(i,j)=0;
  if &pre4(i,j)=-3.5 then &pre4(i,j)=0;
  if &pre5(i,j)=-3.5 then &pre5(i,j)=0;
  if &pre6(i,j)=-3.5 then &pre6(i,j)=0;
  end;
end;

/**initialize amount of months received, by years and total *****/
do i=1 to 19;
  total(i)=0;
end;
totalall=0;

m=0; /** a counter**/

**** calculate the amount of months in total as well in each year receiving AFDC****

do i=1 to 19;
  n=0;    /** a counter**/
  do j= 1 to 12;
    if count(i,j) >=0 then do;
      total(i)=total(i) + count(i,j);
      totalall=totalall + count(i,j);
      n=n+1;
      m=m+1;
    end;
  end;
  if n=0 then total(i)=-4;  /**if event history has a value of -1, -2, -4 dealt with below**/
end;
if m=0 then totalall=-4;

```

```
/**if both -1 and -2 for a respondent, then total=-1*/
do i=1 to 19;
  do j=1 to 12;
    if count(i,j) in (-1,-2,-3) and total(i) ^=-1 then do;
      total(i)=count(i,j);
      totalall=count(i,j);
    end;
  end;
end;

/*correct for invalid skips*/
do i=1 to 19;
  if (pubid=1714 or pubid=6734 or pubid=8405) then total(i)=-3 and totalall=-3;
end;

endsas;
```

NLSY97 Appendix 6:

Event History Creation and Documentation

The NLSY97 survey records significant life-course transitions experienced by young people, such as education, employment, program participation, and marital history, in a longitudinal format. The event history arrays document these events in a chronological format that records the significant transitions in a meaningful manner while maintaining data quality. Using these arrays, researchers can extract the status of a respondent at a point in time or over time. During round 1, event history arrays were generated for four distinct areas: employment, marital/cohabitation status, program participation, and schooling. This section presents information on each type of event history array; for details on the chronological format of the arrays and the naming conventions used to identify the variables, users should refer to appendix 7.

Employment Event History Arrays

In round 1, three employment arrays provide information on the respondent's employment on a weekly basis. These arrays include information about employer jobs only; freelance jobs were not included in the arrays. All employment arrays provide information starting in the week that the respondent turned 14 and ending in the week that he or she was interviewed.

1. EMP_STATUS

This main array presents the employment status of a respondent in a particular week. The codes and their explanations follow:

Code	Definition
Status=0: No information reported to account for week	Not assigned during round 1.
Status=1: Not associated with an employer, not actively searching for an employer job	Refers to weeks during a between-jobs gap in which the respondent is not actively searching and reports working at a freelance job. Since the actual weeks working at a freelance job cannot be determined, all weeks in which the respondent is not actively searching are coded in this manner. Similar information was not collected for the within-job gaps during round 1.
Status=2: Not working (unemployment vs. out of labor force cannot be determined)	Assigned when the respondent is not asked follow-up questions about his or her search activity during a within-job gap or a between-jobs gap.
Status=3: Associated with an employer, periods not working for employer are missing	Used when a respondent reports an indeterminate start or stop date for a within-job gap.
Status=4: Unemployed	Indicates that the respondent reports actively searching for work during a within-job gap or a between-jobs gap. When the number of weeks unemployed do not account for the entire gap period, weeks unemployed are assumed to occur in the middle of that period.
Status=5: Out of the labor force	Assigned during a between-jobs gap or a within-job gap when the respondent is either not actively searching for work or on layoff from a job.
Status=6: Active military service	Indicates that the respondent is tied to the military.
Status=9701 to 9707: Employer on roster	Refers to the employer number on the employer roster (YEMP_UID.01 to YEMP_UID.07). Presence of an employer number indicates that the respondent was working during a given week. Civilian work takes precedence over other activities, such as job search. Respondents who report working at an employer job for one day in a given week are listed as having worked at that job for the entire week, regardless of other activities.

2. EMP_DUAL_JOB#

If a respondent holds more than one employer job during a week, the second employer job is presented in one of the six dual jobs arrays. These arrays contain only the job number of the overlapping job; labor force status information is only included in the main array. For example, if a respondent held two employer jobs (e.g., the first and third jobs listed on the employer roster), during the 52nd week of 1997, the employer number for the first job would be recorded in the EMP_STATUS array and the employer number for the third job would be recorded in the EMP_DUAL_3 array.

3. EMP_HOURS

This final array calculates the total number of hours worked by a respondent at any employer job during a given week. Hours per week worked at each job are assumed constant except during a reported gap, when the hours for that job are assumed to be zero. Each week is assigned a code of '-3 (invalid skip)' when any of the jobs has an indeterminate gap date.

A secondary set of variables translates the reported beginning and ending dates (day, month, and year) of employer jobs and the gaps within those jobs to the week and year naming scheme (e.g., EMP_GAP_START_YEAR.01.01 and EMP_GAP_END_YEAR.01.01 provide the start and end dates of the respondent's first gap in the continuous week and year format). More information about the week and year naming scheme is provided in appendix 7 in this document.

Marital Status Event History Arrays

The NLSY97 marital and cohabitation arrays record changes in the respondent's marital status and cohabitation changes on a monthly basis. The marital/cohabitation history program converts dates reported in the marriage section (beginning and ending dates of cohabitations, marriages, separations, divorces, and widowhoods) to an actual month number, using January 1980 as month #1. Used jointly, these arrays allow the researcher to obtain a detailed history of the respondent's partners and changes in his/her marital and cohabitation status on a monthly basis. All marital/cohabitation arrays provide information beginning in the month that the respondent turned 14 and ending in the month that he or she was interviewed. Additionally, the beginning dates of the youth's first marriage and first cohabitation are provided in two variables: CV_FIRST_MARRY_MONTH and CV_FIRST_COHAB_MONTH.

In round 1, four types of arrays were constructed to record transitions between living without a partner of the opposite sex to cohabiting or to marriage.

1. MAR_STATUS

The main array presents the status (e.g., never married/not cohabiting, cohabiting, married, divorced) of a respondent during a particular month. Marital status takes precedence over cohabiting; for example, if a respondent is divorced and living with another partner, the status listed in this array will be 'divorced.' Respondents who are married but not living with their spouse are coded as married. When a respondent reports an annulment, the marriage is maintained and the marital status code after the annulment is 'divorced.'

2. MAR_COHABITATION

This second array details the partner that the respondent is living with in a particular month. For example, if the respondent is cohabiting, the variable for each month identifies whether the respondent lives with partner 1, partner 2, spouse 1, spouse 2, etc. Users should note that "1" and "2" in this case refer to the respondent's partners/spouses in chronological order. The numbers do not necessarily refer to the same person as the spouse/partner questions asked directly of the respondent during the survey.

If a respondent reports living with two partners in the same month, the first partner is listed in this array, subsequent partners are listed in the MAR_DUAL array.

3. MAR_DUAL

If there is an overlap of partners (e.g., partner 1 leaves at the beginning of the month and partner 2 moves in at the end of the month), this array records the presence of the new partner. The format of these variables is the same as that of the MAR_COHABITATION variables.

4. MAR_PARTNER_LINK

The fourth array links the cohabiting partner or spouse to the partner order in the main survey questions. For example, a number of codebook variables report the status of “Partner #01” or “Partner #02.” If the person the respondent reports cohabiting with in a given month is the first person asked about in the 1997 survey, the MAR_PARTNER_LINK variable for that month is 9701. If the partner in a given month is the second partner asked about in 1997, this variable is coded 9702, and so on. This array allows the researcher to identify characteristics of the respondent’s partner and to link them with spells of marriage or cohabitation.

Program Participation Event History Arrays

Program participation arrays were constructed individually for three programs—AFDC, Food Stamps, and WIC. The AFDC array includes all federal and state programs created under Temporary Assistance to Needy Families (TANF) or any government program for needy families that replaces AFDC. All other programs (e.g., LIHEAP, SSI, other) were combined into a fourth array entitled ‘Other.’ For each program type, three arrays were created for round 1. All program participation arrays provide information starting in the month that the respondent turned 14 and ending in the month that he or she was interviewed.

A secondary set of variables translates the reported beginning and ending dates (month and year) of a spell within the program into the continuous month scheme (e.g., AFDC_START_MONTH and AFDC_STOP_MONTH). More information about the continuous month scheme is provided in appendix 7 in this document.

1. STATUS

The main array, (e.g., AFDC_STATUS), presents the status—receiving or not—of a respondent during each month. When asked for the start or stop date of a spell, the respondent could respond ‘don’t know’ or ‘refuse’ to any component. In this case, the respondent was then asked how many weeks the spell lasted. The number of reported weeks was then divided by 4.3 to determine the equivalent number of months. If a fraction of a month was reported, then the entire month was counted as a month receiving benefits. Using a combination of start date, stop date, and week information, each spell was defined and a value of ‘1’ inserted into the status array to indicate months of receipt. The months that a respondent did not receive that benefit, but was eligible to receive it, have a value of ‘0.’ An edit variable, (e.g., AFDC_EDIT_DATE) flags respondent-reported and imputed dates. The process by which imputed dates and the corresponding edit flag were assigned is described below:

Flag	Definition
Edit Flag=1: Respondent reported participation dates	Respondent reported a complete start and stop date and is not currently receiving. If the respondent reports still receiving at the time of the interview, the interview date is assigned as the temporary stop date. In the next survey round, the respondent will be asked if he or she is still receiving; if not, a permanent stop date equivalent to the previous round's interview date will be assigned. If the respondent reports receiving, participation will continue in filling the array.
Edit Flag=2: Start month imputed	Total weeks known: If the respondent reports not currently receiving, then set the month equal to January and count forward by the number of weeks to imply a stop date. If currently receiving, then count back by the number of weeks from the interview date to impute a start month. If the month indicated by the count falls short of the start year, the start month is December of the start year. If the month occurs in the year before the reported start year, then the start month is January of the start year.
	Total weeks unknown: If the respondent reports not currently receiving, then the start month is set to January. Use December as the stop month and the start year as the stop year. If the respondent reports currently receiving, use December as the start month.
Edit Flag=3: Start month and year imputed	Total weeks known: Count back by the number of weeks from the interview date if currently receiving. If not currently receiving, then count back from interview date to find the most recent year the respondent could have begun receiving and call the start date January of that year; then count forward the number of weeks from that date to imply a stop date.
	Total weeks unknown: If currently receiving, begin the spell at the respondent's 14 th birthday.
Edit Flag=4: Stop month imputed	Total weeks known: If not currently receiving, then count forward from start date. If the month indicated falls short of the stop year, then use January of the stop year as the stop month; if the number of months exceeds the stop year, then set the stop month to December of the stop year. If the stop year is equal to the interview year and the stop month exceeds the interview month, then stop at the interview date.
	Total weeks unknown: If not currently receiving, then use December of stop year for the stop month.
Edit Flag=5: Stop month and year imputed.	Total weeks known: If not still receiving, count forward from the start date.
	Total weeks unknown: If not currently receiving, then use December of the start year as the stop month and the start year as the stop year.
Edit Flag=6: Start and stop dates imputed.	Total weeks unknown: The imputed dates are based on the previous interview's date (start date) to the current interview date (stop date); in round 1, the last interview date is the respondent's 14 th birth month and year.

2. AMOUNT RECEIVED

If a respondent reports receiving in a particular month, a second array presents the amount received in each month (e.g., AFDC_AMT). The dollar values asked about during the interview were meant to be monthly values. However, some responses were higher than the federal or state limits on the amount received from a particular benefit. A likely reason is that the respondent mistakenly reported a total value rather than a monthly value. Values determined to be too high were divided by the number of months the respondent reported receiving the benefit. These values were used in the AMT arrays instead. A second set of edit variables (e.g., AFDC_EDIT_AMT) flags these values for a particular spell. In round 1, the edit values were set for the latest year available. For AFDC, the highest benefit offered was \$1229; reported amounts falling above this maximum were edited. The maximum amount accepted for food stamps was \$936; the edit flag indicates reported amounts that fall above this maximum.

3. HOUSEHOLD MEMBERS RECEIVING

If a respondent reports receiving in a particular month, the persons in the household who benefit from the program in each month (e.g., respondent only, child only, respondent and child) are recorded in a third array (e.g., AFDC_HH). This program condenses the set of answers from the question in the survey that collects this information; for example, see YPRG-18300.01_001 to YPRG-18300.01_005 for AFDC.

IV. Schooling Event History Arrays

In round 1, a set of schooling variables was created for each year beginning in 1980, the year when the first information is available in the survey, to 1997. The education arrays are somewhat different than the other event history arrays. Information on a respondent's education is reported on a yearly basis, rather than monthly or weekly. This approach was used to combine information from the youth questionnaire, which collects more detailed data, and from the parent questionnaire, which presents information only for each year. In general, these variables refer to the school year rather than the calendar year. That is, 1991 in a variable title or in the data for a variable generally indicates the school year starting in fall 1991 and ending in spring 1992.

1. SCH_YEAR_TO_GRADE

This array presents the grade the respondent attended during the school year. The last two digits of the question name indicate the school year. For example, SCH_YEAR_TO_GRADE.90 refers to the grade attended by the respondent during the school year that starts in fall 1990 and ends in spring 1991.

2. SCH_GRADE_TO_YEAR

This array refers to the year the respondent attended a certain grade. For example, if the respondent attended second grade in 1992–93, then SCH_GRADE_TO_YEAR.2 would have the value 1992.

3. SCH_CHANGES

This array counts the number of times the respondent changed the school attended during the school year. For example, SCH_CHANGES.90 shows how many different schools the respondent attended during the school year that started in fall 1990 and ended in spring 1991.

4. SCH_MNTHS_MISSED

This array presents the number of months during the school year that the respondent did not attend school. For example, if SCH_MNTHS_MISSED.90 has a value of 3 for a respondent, then that respondent had a gap in attendance of three months during the school year that started in the fall of 1990 and ended in the spring of 1991. A gap is defined as missing school for one or more months (not including summer vacation); gaps do not have to be consecutive.

5. SCH_SUMMER_SCHOOL

This array refers to extra school classes during an educational break in a given school year, such as summer school. For example, SCH_SUMMER_SCHOOL.90 shows whether the respondent attended school during a break in the 1990–91 school year.

6. SCH_GRADE_PROGRESS

This array has positive values if there are any special events that occurred during the school grade. For example, a positive value in SCH_GRADE_PROGRESS.2 indicates that the respondent was skipped or demoted during second grade. Researchers should note that parents might have been confused as to how to answer the skip grade questions asked during the interview. For example, there are parents who say their child skipped from 5th to 6th grade, while others say from 4th to 6th grades. Both of these cases are probably stating that the child missed most or all of the 5th grade. To resolve this ambiguity, the code states that if a child is skipped consecutive years then the first year (i.e. 5th grade) was missed. If a parent

reports non-consecutive years (i.e. 4th to 6th) then the program assumes the year(s) in the middle are the ones not attended.

7. SCH_YEAR_PROGRESS

This array refers to any special events that occurred during the school year. The question name's last two digits indicate the school year this variable refers to. For example, SCH_YEAR_PROGRESS.90 shows special events that occurred during the school year that starts in fall 1990 and ends in spring 1991. The special events, such as grades skipped or demoted to, are defined in the same way as in the previous array.

8. SCH_SUSPENSIONS

This array counts the number of days during the school year the respondent was suspended from school. For example, if SCH_SUSPENSIONS.90 has a value of 3 then the respondent was suspended from school 3 days during the school year that started in fall 1990 and ended in spring 1991.

NLSY97 Appendix 7:

Continuous Month Scheme and Crosswalk

Continuous Month Timeline

The non-employment event history arrays are presented using two different timelines. The first, a continuous month approach, labels January 1980 as month 1, February 1980 as month 2, and so on. Thus, a respondent born in month 4 might start receiving public assistance in month 193 and leave the program in month 198. Key events occurring during the life of each cohort member can be indexed within this month-by-month structure. To aid users, a number of variables other than the event history arrays have already been created using the event history format. For example, the month of the respondent's birth and the month of the respondent's interview are both presented in continuous month format. Second, the event history arrays can be constructed using actual month and year dates. Using this approach, the same respondent, born in April 1980, started receiving public assistance in January 1997 and left the program in June 1997.

The data set includes variables for the event history arrays for every month number under the continuous month scheme. However, actual month and year dates are only included for a given respondent in months in which transitions occurred. That is, the respondent in the example above would have continuous month variables relating to program participation status for every month beginning with 172 (the month of the respondent's 14th birthday) but would only have actual date variables for January and June 1997. To aid users in moving between the two dating schemes, table 1 in this appendix contains a crosswalk between actual dates and continuous month numbers.

The continuous month system allows researchers to easily compare the times at which various events occurred in the youth's life. The following example illustrates a case in which a youth turns 17 in February 1997 and, in the same month (month 194 in the continuous timeline), begins receiving AFDC. She remains on AFDC for 2 months (month 194 and month 195). The following table illustrates this information using a month-by-month timeline.

Non-Employment Event History Array Incorporating the Month-by-Month Timeline

Question Name	Participation Status	Month Number (created by the user)	Birth date: CV_CHILD_BIRTH_MONTH
AFDC_STATUS.97.01	0	193	
AFDC_STATUS.97.02	1	194	194
AFDC_STATUS.97.03	1	195	
AFDC_STATUS.97.04	0	196	

In addition linking the dates through the crosswalk, any user who selects an event array will be able to extract from the CD additional characteristics that relate to that event. The following table presents the additional information that a user may chose to link to the non-employment event history arrays. A user is able to create these and other variable combinations for any status array.

Status:	Variable 1	Variable 2	Variable 3
Enrollment	Highest grade attended	Highest grade completed	Highest degree earned
Cohabitation/Marital	Partner's race	Partner's religion	Partner's grade level
AFDC	Received during month	Benefit amount	People benefiting

Continuous Week Timeline

In the event history section of the CD-ROM, an employment history of the number of weeks worked is presented in a continuous week-by-week status array. This array is very similar to the month-by-month scheme used for the rest of the event history variables. In this format, the first week of January 1980 is number week 1, the second week of January 1980 is numbered week 2, and so on; weeks are listed by

exact date as well. This week-by-week array lists each respondent's employment status for each week since age 14.

As with the month-by-month arrays, respondents have data available for every week in the continuous week arrays but only for transition weeks in the actual date arrays. However, users should note that, due to technical considerations, the variable titles in the data number weeks by calendar year rather than from week 1 through week 992. For example, a variable might read "Employment: Employment Status in Week 07 Year 94" instead of "Employment: Employment Status in Week 737." Table 2 in this appendix provides a crosswalk between the actual beginning date of each week, the continuous week number, and the week number for each week in a calendar year. This table begins with January 1994 because this is the first week in which any NLSY97 respondent could have reached age 14, the youngest age for which the employemnt event history variables were created. Users who need to see earlier continuous week numbers should contact NLS User Services.

Naming Conventions

The question names of the NLSY97 event history variables incorporate the dates to which the variables apply. The marital status and program participation array titles include the month and year (e.g., AFDC_AMT.97.12 corresponds to December 1997). Likewise, employment array variables are listed for each week and year (e.g., EMP_STATUS.52.97 corresponds to the 52nd week in 1997). The schooling variables are yearly and employ a slightly different system. In general, these variables refer to the school year rather than the calendar year. That is, 1991 in a variable title or in the data for a variable generally indicates the school year starting in fall 1991 and ending in spring 1992.

Table 1. Continuous Month Crosswalk

Month and year	Continuous month number	Month and year	Continuous month number
January 1980	1	July 1983	43
February 1980	2	August 1983	44
March 1980	3	September 1983	45
April 1980	4	October 1983	46
May 1980	5	November 1983	47
June 1980	6	December 1983	48
July 1980	7	January 1984	49
August 1980	8	February 1984	50
September 1980	9	March 1984	51
October 1980	10	April 1984	52
November 1980	11	May 1984	53
December 1980	12	June 1984	54
January 1981	13	July 1984	55
February 1981	14	August 1984	56
March 1981	15	September 1984	57
April 1981	16	October 1984	58
May 1981	17	November 1984	59
June 1981	18	December 1984	60
July 1981	19	January 1985	61
August 1981	20	February 1985	62
September 1981	21	March 1985	63
October 1981	22	April 1985	64
November 1981	23	May 1985	65
December 1981	24	June 1985	66
January 1982	25	July 1985	67
February 1982	26	August 1985	68
March 1982	27	September 1985	69
April 1982	28	October 1985	70
May 1982	29	November 1985	71
June 1982	30	December 1985	72
July 1982	31	January 1986	73
August 1982	32	February 1986	74
September 1982	33	March 1986	75
October 1982	34	April 1986	76
November 1982	35	May 1986	77
December 1982	36	June 1986	78
January 1983	37	July 1986	79
February 1983	38	August 1986	80
March 1983	39	September 1986	81
April 1983	40	October 1986	82
May 1983	41	November 1986	83
June 1983	42	December 1986	84

Table 1. Continuous Month Crosswalk (Continued)

Month and year	Continuous month number	Month and year	Continuous month number
January 1987	85	July 1990	127
February 1987	86	August 1990	128
March 1987	87	September 1990	129
April 1987	88	October 1990	130
May 1987	89	November 1990	131
June 1987	90	December 1990	132
July 1987	91	January 1991	133
August 1987	92	February 1991	134
September 1987	93	March 1991	135
October 1987	94	April 1991	136
November 1987	95	May 1991	137
December 1987	96	June 1991	138
January 1988	97	July 1991	139
February 1988	98	August 1991	140
March 1988	99	September 1991	141
April 1988	100	October 1991	142
May 1988	101	November 1991	143
June 1988	102	December 1991	144
July 1988	103	January 1992	145
August 1988	104	February 1992	146
September 1988	105	March 1992	147
October 1988	106	April 1992	148
November 1988	107	May 1992	149
December 1988	108	June 1992	150
January 1989	109	July 1992	151
February 1989	110	August 1992	152
March 1989	111	September 1992	153
April 1989	112	October 1992	154
May 1989	113	November 1992	155
June 1989	114	December 1992	156
July 1989	115	January 1993	157
August 1989	116	February 1993	158
September 1989	117	March 1993	159
October 1989	118	April 1993	160
November 1989	119	May 1993	161
December 1989	120	June 1993	162
January 1990	121	July 1993	163
February 1990	122	August 1993	164
March 1990	123	September 1993	165
April 1990	124	October 1993	166
May 1990	125	November 1993	167
June 1990	126	December 1993	168

Table 1. Continuous Month Crosswalk (Continued)

Month and year	Continuous month number	Month and year	Continuous month number
January 1994	169	July 1996	199
February 1994	170	August 1996	200
March 1994	171	September 1996	201
April 1994	172	October 1996	202
May 1994	173	November 1996	203
June 1994	174	December 1996	204
July 1994	175	January 1997	205
August 1994	176	February 1997	206
September 1994	177	March 1997	207
October 1994	178	April 1997	208
November 1994	179	May 1997	209
December 1994	180	June 1997	210
January 1995	181	July 1997	211
February 1995	182	August 1997	212
March 1995	183	September 1997	213
April 1995	184	October 1997	214
May 1995	185	November 1997	215
June 1995	186	December 1997	216
July 1995	187	January 1998	217
August 1995	188	February 1998	218
September 1995	189	March 1998	219
October 1995	190	April 1998	220
November 1995	191	May 1998	221
December 1995	192	June 1998	222
January 1996	193	July 1998	223
February 1996	194	August 1998	224
March 1996	195	September 1998	225
April 1996	196	October 1998	226
May 1996	197	November 1998	227
June 1996	198	December 1998	228

Appendix 7: Continuous Month Scheme and Crosswalk

Table 2. Continuous Week Crosswalk

Week start date	Continuous week number	Calendar year week number	Week start date	Continuous week number	Calendar year week number
12/26/93	731	1	10/30/94	775	45
01/02/94	732	2	11/06/94	776	46
01/09/94	733	3	11/13/94	777	47
01/16/94	734	4	11/20/94	778	48
01/23/94	735	5	11/27/94	779	49
01/30/94	736	6	12/04/94	780	50
02/06/94	737	7	12/11/94	781	51
02/13/94	738	8	12/18/94	782	52
02/20/94	739	9	12/25/94	783	53
02/27/94	740	10	01/01/95	784	1
03/06/94	741	11	01/08/95	785	2
03/13/94	742	12	01/15/95	786	3
03/20/94	743	13	01/22/95	787	4
03/27/94	744	14	01/29/95	788	5
04/03/94	745	15	02/05/95	789	6
04/10/94	746	16	02/12/95	790	7
04/17/94	747	17	02/19/95	791	8
04/24/94	748	18	02/26/95	792	9
05/01/94	749	19	03/05/95	793	10
05/08/94	750	20	03/12/95	794	11
05/15/94	751	21	03/19/95	795	12
05/22/94	752	22	03/26/95	796	13
05/29/94	753	23	04/02/95	797	14
06/05/94	754	24	04/09/95	798	15
06/12/94	755	25	04/16/95	799	16
06/19/94	756	26	04/23/95	800	17
06/26/94	757	27	04/30/95	801	18
07/03/94	758	28	05/07/95	802	19
07/10/94	759	29	05/14/95	803	20
07/17/94	760	30	05/21/95	804	21
07/24/94	761	31	05/28/95	805	22
07/31/94	762	32	06/04/95	806	23
08/07/94	763	33	06/11/95	807	24
08/14/94	764	34	06/18/95	808	25
08/21/94	765	35	06/25/95	809	26
08/28/94	766	36	07/02/95	810	27
09/04/94	767	37	07/09/95	811	28
09/11/94	768	38	07/16/95	812	29
09/18/94	769	39	07/23/95	813	30
09/25/94	770	40	07/30/95	814	31
10/02/94	771	41	08/06/95	815	32
10/09/94	772	42	08/13/95	816	33
10/16/94	773	43	08/20/95	817	34
10/23/94	774	44	08/27/95	818	35

Table 2. Continuous Week Crosswalk (Continued)

Week start date	Continuous week number	Calendar year week number	Week start date	Continuous week number	Calendar year week number
09/03/95	819	36	07/07/96	863	28
09/10/95	820	37	07/14/96	864	29
09/17/95	821	38	07/21/96	865	30
09/24/95	822	39	07/28/96	866	31
10/01/95	823	40	08/04/96	867	32
10/08/95	824	41	08/11/96	868	33
10/15/95	825	42	08/18/96	869	34
10/22/95	826	43	08/25/96	870	35
10/29/95	827	44	09/01/96	871	36
11/05/95	828	45	09/08/96	872	37
11/12/95	829	46	09/15/96	873	38
11/19/95	830	47	09/22/96	874	39
11/26/95	831	48	09/29/96	875	40
12/03/95	832	49	10/06/96	876	41
12/10/95	833	50	10/13/96	877	42
12/17/95	834	51	10/20/96	878	43
12/24/95	835	52	10/27/96	879	44
12/31/95	836	1	11/03/96	880	45
01/07/96	837	2	11/10/96	881	46
01/14/96	838	3	11/17/96	882	47
01/21/96	839	4	11/24/96	883	48
01/28/96	840	5	12/01/96	884	49
02/04/96	841	6	12/08/96	885	50
02/11/96	842	7	12/15/96	886	51
02/18/96	843	8	12/22/96	887	52
02/25/96	844	9	12/29/96	888	1
03/03/96	845	10	01/05/97	889	2
03/10/96	846	11	01/12/97	890	3
03/17/96	847	12	01/19/97	891	4
03/24/96	848	13	01/26/97	892	5
03/31/96	849	14	02/02/97	893	6
04/07/96	850	15	02/09/97	894	7
04/14/96	851	16	02/16/97	895	8
04/21/96	852	17	02/23/97	896	9
04/28/96	853	18	03/02/97	897	10
05/05/96	854	19	03/09/97	898	11
05/12/96	855	20	03/16/97	899	12
05/19/96	856	21	03/23/97	900	13
05/26/96	857	22	03/30/97	901	14
06/02/96	858	23	04/06/97	902	15
06/09/96	859	24	04/13/97	903	16
06/16/96	860	25	04/20/97	904	17
06/23/96	861	26	04/27/97	905	18
06/30/96	862	27	05/04/97	906	19

Appendix 7: Continuous Month Scheme and Crosswalk

Table 2. Continuous Week Crosswalk (Continued)

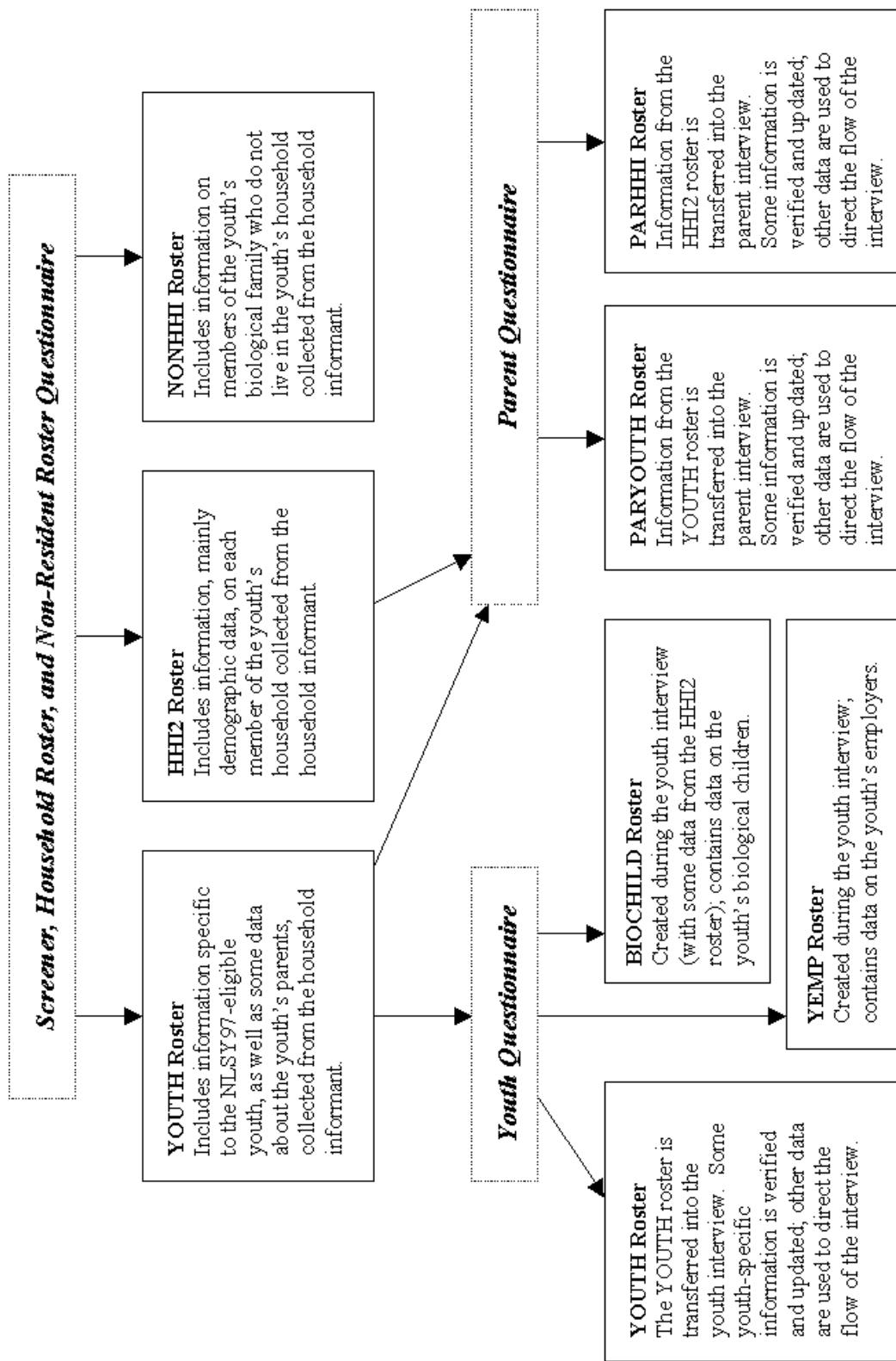
Week start date	Continuous week number	Calendar year week number	Week start date	Continuous week number	Calendar year week number
05/11/97	907	20	03/08/98	950	11
05/18/97	908	21	03/15/98	951	12
05/25/97	909	22	03/22/98	952	13
06/01/97	910	23	03/29/98	953	14
06/08/97	911	24	04/05/98	954	15
06/15/97	912	25	04/12/98	955	16
06/22/97	913	26	04/19/98	956	17
06/29/97	914	27	04/26/98	957	18
07/06/97	915	28	05/03/98	958	19
07/13/97	916	29	05/10/98	959	20
07/20/97	917	30	05/17/98	960	21
07/27/97	918	31	05/24/98	961	22
08/03/97	919	32	05/31/98	962	23
08/10/97	920	33	06/07/98	963	24
08/17/97	921	34	06/14/98	964	25
08/24/97	922	35	06/21/98	965	26
08/31/97	923	36	06/28/98	966	27
09/07/97	924	37	07/05/98	967	28
09/14/97	925	38	07/12/98	968	29
09/21/97	926	39	07/19/98	969	30
09/28/97	927	40	07/26/98	970	31
10/05/97	928	41	08/02/98	971	32
10/12/97	929	42	08/09/98	972	33
10/19/97	930	43	08/16/98	973	34
10/26/97	931	44	08/23/98	974	35
11/02/97	932	45	08/30/98	975	36
11/09/97	933	46	09/06/98	976	37
11/16/97	934	47	09/13/98	977	38
11/23/97	935	48	09/20/98	978	39
11/30/97	936	49	09/27/98	979	40
12/07/97	937	50	10/04/98	980	41
12/14/97	938	51	10/11/98	981	42
12/21/97	939	52	10/18/98	982	43
12/28/97	940	1	10/25/98	983	44
01/04/98	941	2	11/01/98	984	45
01/11/98	942	3	11/08/98	985	46
01/18/98	943	4	11/15/98	986	47
01/25/98	944	5	11/22/98	987	48
02/01/98	945	6	11/29/98	988	49
02/08/98	946	7	12/06/98	989	50
02/15/98	947	8	12/13/98	990	51
02/22/98	948	9	12/20/98	991	52
03/01/98	949	10	12/27/98	992	1

NLSY97 Appendix 8:
Round 1 Instrument Rosters

During the course of the CAPI survey, a number of rosters, or matrices of data, are constructed. These rosters contain one or more pieces of information on a given subject. Rosters are often presented to the interviewers as lists of information that are used to verify information, or from which one of the subjects on the roster is chosen as the answer to a survey question. For example, the EMPLOYER roster (a list of employers for whom the respondent has worked since his or her 14th birthday) is used to verify that the list of employers is accurate. Many of the rosters used during the administration of the survey are presented as consolidated blocks of data on the public release CD-ROM.

In round 1, a number of rosters were used to organize information from the Screener, Household Roster, and Non-Resident Roster Questionnaire. Some of this information was then transferred into the parent and youth interviews for verification and for use in determining question paths. Figure 1 below identifies the key rosters in the round 1 survey and shows how they were used in different parts of the survey.

The following pages list the variable names, titles, and reference numbers for the various instrument rosters used during the round 1 interview. These lists are intended to aid researchers in identifying the types of information organized in each roster and to better follow the flow of information through the interview.



HHI2 Roster (Household enumeration)

PREFIX = "HHI2" (e.g., HHI2_AGE.01)

Variable Name	Title: All end in (Scr Ros Item)	Reference #
_AGE.xx	Age of HH Member xx as of Intdate	R10803.–R10818.
_ASVAB.xx	Was HH Member xx Flagged for ASVAB	R10819.–R10834.
_DOB.xx	Date of Birth of HH Member xx	R10851.–R10865.02
_DADID.xx	ID Number of HH Member xx Bio Dad	R10835.–R10850.
_EMPLOYED.xx	Employment Status of HH Member xx	R10898.–R10913.
_ENROLLNEXT.xx	Will HH Member xx Be Enrolled Next Fall	R10914.–R10929.
_ENROLLSTAT.xx	Is HH Member xx Currently Enrolled	R10930.–R10945.
_ETHNICITY.xx	Is HH Member xx Hispanic?	R10946.–R10961.
_HHI1ID.xx	ID of HH Member xx from HHI1 Roster	R10978.–R10993.
_HIGHGRADE.xx	HH Member xx Highest Grade Completed	R10994.–R11009.
_HHID.xx	HH Member xx ID	R11010.–R11025.
_INFORMANT.xx	Is HH Member xx the Informant	R11026.–R11041.
_MARSTAT.xx	HH Member xx Marital Status	R11042.–R11057.
_MOMID.xx	HH Member xx Bio Moms ID	R11058.–R11073.
_PARTNER.xx	HH Member xx Have a Partner	R11122.–R11137.
_PARTNERID.xx	ID of HH Member xx Partner	R11138.–R11153.
_RACE.xx	Race of HH Member xx	R11154.–R11169.
_REL1.xx	Relationship of Person 1 to HH Member xx	R11170.–R11185.
_REL2.xx	Relationship of Person 2 to HH Member xx	R11319.–R11334.
_REL3.xx	Relationship of Person 3 to HH Member xx	R11342.–R11357.
_REL4.xx	Relationship of Person 4 to HH Member xx	R11358.–R11373.
_REL5.xx	Relationship of Person 5 to HH Member xx	R11374.–R11389.
_REL6.xx	Relationship of Person 6 to HH Member xx	R11390.–R11405.
_REL7.xx	Relationship of Person 7 to HH Member xx	R11406.–R11421.
_REL8.xx	Relationship of Person 8 to HH Member xx	R11422.–R11437.
_REL9.xx	Relationship of Person 9 to HH Member xx	R11438.–R11453.
_REL10.xx	Relationship of Person 10 to HH Member xx	R11186.–R11201.
_REL11.xx	Relationship of Person 11 to HH Member xx	R11202.–R11217.
_REL12.xx	Relationship of Person 12 to HH Member xx	R11218.–R11233.
_REL13.xx	Relationship of Person 13 to HH Member xx	R11234.–R11249.
_REL14.xx	Relationship of Person 14 to HH Member xx	R11250.–R11265.
_REL15.xx	Relationship of Person 15 to HH Member xx	R11266.–R11281.
_REL16.xx	Relationship of Person 16 to HH Member xx	R11282.–R11297.
_REVDOLEL.xx	Is HH Member xx DOL Eligible (Revised)	R11454.–R11469.
_REVETPEL.xx	Is HH Member xx ETP Eligible (Revised)	R11470.–R11485.
_REVSTPEL.xx	Is HH Member xx STP Eligible (Revised)	R11486.–R11501.
_SEX.xx	Gender of HH Member xx	R11502.–R11517.
_SPECIAL.xx	HH Member xx Approved Special Accommodations	R11518.–R11533.
_SOPARID.xx	ID of HH Member xx Spouse or Partner	R11541.–R11556.
_SPOUSEID.xx	ID of HH Member xx Spouse in HH	R11557.–R11572.
_UID.xx	HH Member xx Unique ID	R11621.–R11636.

Appendix 8: Round 1 Instrument Rosters

NONHHI Roster (Non-resident relative enumeration)

PREFIX = "NONHHI" (e.g., NONHHI_AGE.01)

Variable Name	Title: All end in (Scr Ros Item)	Reference #
_AGE.xx	Age Non-Res Member xx at Interview Date, Youth #1	R11637.–R11659.
_DECEASED.xx	Is Non-Res Member xx Deceased, Youth #1	R11660.–R11682.
_DEGREE.xx	Non-Res Member xx Have a Degree, Youth #1	R11683.–R11703.
_EMPLOYED.xx	Employment Status of Non-Res Member xx, Youth #1	R11704.–R11724.
_ETHNICITY.xx	Is Non-Res Member xx Hispanic, Youth #1	R11725.–R11745.
_HIGHGRADE.xx	HGC by Non-Res Member xx, Youth #1	R11769.–R11789.
_MARSTAT.xx	Marital Status of Non-Res Member xx, Youth #1	R11799.–R11821.
_RACE.xx	Race of Non-Res Member xx, Youth #1	R11845.–R11865.
_RELATION.xx	Relationship of Non-Res Member xx to Youth #1	R11866.–R11883.
_SEX.xx	Gender of Non-Res Member xx, Youth #1	R11884.–R11906.
_UID.xx	Unique ID of Non-Res Member xx, Youth #1	R11907.–R11929.

BIOCHILD Roster (Youth's children)

PREFIX = "BIOCHILD" (e.g., BIOCHILD_BDATE.01)

Variable Name	Title: All end in (Ros Item)	Reference #
_BDATE.xx	Birthdate of R Bio Child xx, Final	R05202.–R05203.02
_DEAD.xx	Is R Bio Child xx Deceased, Final	R05204.–R05205.
_RESIDE.xx	Does R Bio Child xx Reside in Household, Final	R05212.–R05213.
_SEX.xx	Gender of R Bio Child xx, Final	R05214.–R05215.
_ID.xx	ID Number of Biochild xx	R05216.–R05217.

YEMP Roster (Youth's employers)

PREFIX = "YEMP" (e.g., YEMP_CURFLAG.01)

Variable Name	Title: All end in (Ros Item)	Reference #
_CURFLAG.xx	Was R Currently Employed at Interview Date Job xx	R05252.–R05258.
_INTERN.xx	Is This an Internship Employer Job xx	R05265.–R05271.
_STARTDATE.xx	Employer Start Month/Day/Year Job xx	R05297.–R05303.02
_STOPDATE.xx	Employer Stop Month/Day/Year Job xx	R05304.–R05310.02
_UID.xx	Employer Unique ID Number Job xx	R05311.–R05317.

YOUTH Roster (Youth respondent information)
 PREFIX = "YOUTH" (e.g., YOUTH_ADOPDADID.01)

Variable Name	Title: All end in (Ros Item)	Reference #
_ADOPDADID.01	R 01 Adoptive Dads ID	R05318.
_ADOPMOMID.01	R 01 Adoptive Moms ID	R05319.
_BOTHBIO.01	Does R 01 Live with Both Bio Parents?	R05322.
_DADID.10	R 01 Bio Dads ID	R05323.
_EMANCIPAT.01	Is R 01 Emancipated?	R05326.
_FOSTDADID.01	R 01 Foster Dads ID	R05327.
_FOSTMOMID.01	R 01 Foster Moms ID	R05328.
_GRADE.01	R 01 Current Grade	R05329.
_HHADOPTKID.01	Does R 01 Have Any Adopted Kids in HH?	R05330.
_HHBIOKID.01	Does R 01 Have Any Bio Kids in HH?	R05331.
_HHID.01	R 01 HH Id Number	R05332.
_HHSTEPKID.01	Does R 01 Have Any Step Kids?	R05333.
_ID.01	R 01 ID Number	R05334.
_MOMID.01	R 01 Bio Moms ID	R05336.
_NONR1DEAD.01	Is R 01 1 st Non-Resp Bio Parent Deceased?	R05338.
_NONR1ID.01	ID of R 01 1 st Non-Resp Bio Parent	R05339.
_NONR1INHH.01	Does 1 st Non-Resp Bio Parent of R 01 Live in HH?	R05340.
_NONR1SEX.01	Gender of R 01 1 st Non-Resp Bio Parent	R05342.
_NONR2DEAD.01	Is R 01 2 nd Non-Resp Bio Parent Deceased?	R05343.
_NONR2ID.01	R 01 2 nd Non-Resp Bio Parents ID	R05344.
_NONR2INHH.01	Is R 01 2 nd Non-Resp Parent in HH	R05345.
_NONR2SEX.01	Gender of R 01 2 nd Non-Resp Bio Parent	R05347.
_NRADOPTKID.01	Does R 01 Have Any Non-Resident Adopted Kids	R05348.
_NRBIOKID.01	Does R 01 Have Any Non-Resident Bio Kids?	R05349.
_NRDADID.01	ID of R 01 Non-Resident Bio Dad	R05350.
_NRMOMID.01	ID of R 01 Non-Resident Bio Mom	R05351.
_NRSTEPKID.01	Does R 01 Have Any Non-Resident Step Kids	R05352.
_PARENT.01	Relationship of Resp Parent to R 01	R05353.
_PARENTGUAR.01	Does R 01 Have a Resp Parent or Guardian in HH	R05354.
_PARENTID.01	ID of R 01 Resp Parent	R05355.
_PARENTSEX.01	Gender of R 01 Resp Parent	R05356.
_SOPARID.01	ID of R 01 Spouse or Partner	R05358.
_STEPDADID.01	ID of R 01 Step Dad	R05359.
_STEPMOMID.01	ID of R 01 Step Mom	R05360.

Appendix 8: Round 1 Instrument Rosters

PARHHI Roster (Household information in parent interview)

PREFIX = "PARHHI" (e.g., PARHHI_AGE.01)

Variable Name	Title: All end in (Par Ros Item)	Reference #
_AGE.xx	Age of HH Member xx as of Interview Date	R06945.–R06953.
_AGEDOL.xx	Age of HH Member xx as of 12/31/1996	R06954.–R06962.
_DADID.xx	Member xx Bio Dads ID	R06963.–R06971.
_DOB.xx	Member xx Date of Birth	R06972.–R06980.02
_DOLEL.xx	Is HH Member xx DOL Eligible (Preliminary)	R06981.–R06989
_ELIGIBLE.xx	Member xx DOL, ETP or STP Eligible	R06990.–R06998.
_EMPLOYED.xx	Employment Status of HH Member xx	R06999.–R07001.
_ENROLLSTAT.xx	Is HH Member xx Currently Enrolled	R07002.–R07009.
_ETPEL.xx	Is HH Member xx ETP Eligible (Preliminary)	R07010.–R07018.
_GRADE.xx	Member xx Current Grade	R07019.–R07027.
_HIGHGRADE.xx	Member xx Highest Grade Completed	R07028.–R07036.
_ID.xx	ID of HH Member xx	R07037.–R07045.
_MARSTAT.xx	Member xx Marital Status	R07046.–R07054.
_MOB.xx	Member xx Month of Birth	R07055.–R07063.
_MOMID.xx	Member xx Bio Moms ID	R07064.–R07072.
_PARTNER.xx	Member xx Have a Partner?	R07082.–R07090.
_RACE.xx	Race of HH Member xx	R07091.–R07099.
_REL1.xx	Relationship of Person 1 to HH Member xx	R07100.–R07108.
_REL10.xx	Relationship of Person 10 to HH Member xx	R07109.–R07117.
_REL11.xx	Relationship of Person 11 to HH Member xx	R07118.–R07126.
_REL12.xx	Relationship of Person 12 to HH Member xx	R07127.–R07135.
_REL13.xx	Relationship of Person 13 to HH Member xx	R07136.–R07144.
_REL14.xx	Relationship of Person 14 to HH Member xx	R07145.–R07153.
_REL15.xx	Relationship of Person 15 to HH Member xx	R07154.–R07162.
_REL16.xx	Relationship of Person 16 to HH Member xx	R07163.–R07167.
_REL17.xx	Relationship of Person 17 to HH Member xx	R07168.–R07172.
_REL18.xx	Relationship of Person 18 to HH Member xx	R07173.–R07177.
_REL19.xx	Relationship of Person 19 to HH Member xx	R07178.–R07182.
_REL2.xx	Relationship of Person 2 to HH Member xx	R07183.–R07191.
_REL20.xx	Relationship of Person 20 to HH Member xx	R07192.–R07196.
_REL3.xx	Relationship of Person 3 to HH Member xx	R07197.–R07205.
_REL4.xx	Relationship of Person 4 to HH Member xx	R07206.–R07214.
_REL5.xx	Relationship of Person 5 to HH Member xx	R07215.–R07223.
_REL6.xx	Relationship of Person 6 to HH Member xx	R07224.–R07232.
_REL7.xx	Relationship of Person 7 to HH Member xx	R07233.–R07241.
_REL8.xx	Relationship of Person 8 to HH Member xx	R07242.–R07250.
_REL9.xx	Relationship of Person 9 to HH Member xx	R07251.–R07259.
_REVDOLEL.xx	Is HH Member xx DOL Eligible (Revised)	R07260.–R07268.
_REVETPEL.xx	Is HH Member xx ETP Eligible (Revised)	R07269.–R07277.
_REVSTPEL.xx	Is HH Member xx STP Eligible (Revised)	R07278.–R07286.
_SEX.xx	Member xx Sex	R07287.–R07295.
_SPOPARID.xx	ID of HH Member xx Spouse or Partner	R07296.–R07304.
_STPEL.xx	Is Member xx STP Eligible (Preliminary)	R07305.–R07313.

PARYOUTH Roster (Youth information for parent interview)

PREFIX = "PARYOUTH" (e.g., PARYOUTH_AGE.01)

Variable Name	Title: All end in (Par Ros Item)	Reference #
_ADOPDADID	Rs Adoptive Dads ID	R07314.
_ADOPMOMID	Rs Adoptive Moms ID	R07315.
_AGE	Age of R as of Interview Date	R07316.
_AGEDOL	Age of R as of 12/31/96	R07317.
_BOTHBIO	Does R Live with Both Bio Parents?	R07318.
_DADID	Rs Bio Dads ID	R07319.
_DOB	Date of Rs Birth	R07320.-R07320.02
_ELIGIBLE	Is R Eligible for DOL, ETP or STP?	R07321.
_EMANCIPAT	Is R Emancipated?	R07322.
_FOSTDADID	Rs Foster Dads ID	R07323.
_FOSTMOMID	Rs Foster Moms ID	R07324.
_GRADE	Rs Current Grade	R07325.
_HHADOPTKID	Does R Have Any Adopted Kids in HH?	R07326.
_HHBIOKID	Does R Have Any Bio Kids in HH?	R07327.
_HHID	Rs HH ID Number	R07328.
_HHSTEPKID	Does R Have Any Step Kids?	R07329.
_ID	Rs ID Number	R07330.
_MARSTAT	Rs Marital Status	R07331.
_MOMID	Rs Bio Moms ID	R07332.
_NONR1DEAD	Is Rs 1 st Non-Resp Bio Parent Deceased?	R07333.
_NONR1ID	ID of Rs 1 st Non-Resp Bio Parent	R07334.
_NONR1INHH	1 st Non-Resp Bio Parent of R Live in HH?	R07335.
_NONR1SEX	Gender of Rs 1 st Non-Resp Bio Parent	R07337.
_NONR2DEAD	Is Rs 2 nd Non-Resp Bio Parent Deceased?	R07338.
_NONR2ID	Rs 2 nd Non-Resp Bio Parents ID	R07339.
_NONR2INHH	Is Rs 2 nd Non-Resp Bio Parent In HH	R07340.
_NONR2SEX	Gender of Rs 2 nd Non-Resp Bio Parent	R07342.
_NRADOPTKID	Does R Have Any Non-Resident Adopted Kids	R07343.
_NRBIOKID	Does R Have Any Non-Resident Bio Kids?	R07344.
_NRDADID	ID of Rs Non-Resident Bio Dad	R07345.
_NRMOMID	ID of Rs Non-Resident Bio Mom	R07346.
_NRSTEPKID	Does R Have Any Non-Resident Step Kids	R07347.
_PARENT	Relationship of Resp Parent to R	R07348.
_PARENTGUAR	R Have a Resp Parent or Guardian in HH	R07349.
_PARENTID	ID of Rs Resp Parent	R07350.
_PARENTSEX	Gender of Rs Resp Parent	R07351.
_SEX	Gender of R	R07352.
_SOPARID	ID of Rs Spouse or Partner	R07353.
_STEPDADID	ID of Rs Stepdad	R07354.
_STEPMOMID	ID of Rs Stepmom	R07355.